An Interop Publication

ConneXions®

The Interoperability Report

May 1996 | *Special Issue: Network Management Today* | Volume 10, No. 5

ConneXions —
The Interoperability Report tracks current and emerging standards and technologies within the computer and communications industry.

In this issue:

ConneXions is published monthly by Interop Company, a division of SOFTBANK Exposition and Conference Company, 303 Vintage Park Drive, Foster City, California, 94404–1138, USA.
Phone: +1 (415) 578-6900
Fax: +1 (415) 525-0194
E-mail: connexions@interop.com

Subscription hotline: 1-800-575-5717
or +1 610-892-1959

ISSN 0894–5926

From the Editor

The *Simple Network Management Protocol* (SNMP), was introduced in 1988. The initial version (SNMPv1) is widely implemented, deployed, and used. But like most technologies in the area of networking, SNMP has undergone revisions and updates. The new version, SNMPv2, is now in the final stages of standardization. In this issue we will look at the current state of Internet standard management from both a practical and theoretical perspective.

We begin with an article about the InteropNet. When we first introduced the concept of a "Show-and-Telnet" in 1988, it consisted of only a few hundred feet of thick Ethernet, interconnecting the fifty or so exhibitors. Since that time, the InteropNet has grown to encompass the full spectrum of networking technologies, including some of the latest "bleeding edge" systems. Its major design criteria: flexibility, interoperability, modularity, and transportability allows the network to move from venue to venue on the NetWorld+Interop World Tour. Once the network is in place at a given location, it forms a critical component of the trade show that allows both visitors and exhibitors to access a variety of systems on the show floor and elsewhere. Managing this network is the job of the "NOC Team," a dedicated group of networking experts from around the globe. We asked two of them, Bobby Krupczak and Steve Hultquist, to describe how the InteropNet is managed.

One of the most important weaknesses of SNMPv1 is the lack of adequate mechanisms for securing the management function. This includes authentication and privacy, as well as an administrative framework for authorization and access control. The IETF working group that developed SNMPv2 wanted to include security functionality in the new version. Unfortunately, the working group was not able to reach agreement on *how* to define the required security mechanisms, and so *two* rather than one proposals were put forward. These proposals, known as SNMPv2u and SNMPv2*, are described in two articles, the first by Glenn Waters and the second by David Partain. Most experts agree that having two sets of SNMP security standards is not a long-term solution, so consider these articles snapshots of a technology under development. We will report on the ultimate outcome in a future issue.

Our final article is an overview of RMON2, the latest version of the *Remote Network Monitoring* MIB. Like SNMP itself, the RMON standards have evolved and now include the ability to monitor protocol traffic above the MAC level. The article is by Bill Stallings.

Managing the InteropNet™

by

Bobby Krupczak, Empire Technologies, Inc.

and

Steve Hultquist, Worldwide Solutions, Inc.

Introduction

The NetWorld+Interop conference and trade show has grown from a small gathering of industry professionals interested in testing and evaluating their networking hardware and software into a full-blown industry event attended by over 50,000 people. At the heart of the show is the *InteropNet,* which interconnects the exhibitors, attendees, remote sites, and the Internet. The InteropNet emulates the complex networks found in large corporations and educational institutions. Furthermore, it often mirrors (if not magnifies) the challenges faced today when building such networks. NetWorld+Interop attendees can see, test, and inspect new and emerging network technologies and products functioning on a real, live network. The interconnection of all exhibitors, for instance, provides the opportunity for exhibitors to prove interoperability with other exhibitors during the show. The challenge of designing, building, and installing the InteropNet is immense. The task of managing its operation during the show is as well.

The InteropNet presents many unique and equally immense constraints and challenges due to its size, the fact that it must travel from city to city, the desire to incorporate new and emerging technologies, the lack of control over the equipment that is connected, and its dynamic composition. Although the InteropNet is more complex in size and scope, the experiences gained from it are most certainly applicable. An enormous amount of work goes into bringing the InteropNet to life, much of which goes unrecognized. The design, assembly, and deployment of a network this size would constitute at least three separate articles in and of itself.

The goal of this article is to articulate only one of those aspects of how we, the *Network Operations Center* (NOC) team, collectively manage the InteropNet. We use the term "manage" somewhat loosely so as to incorporate the full spectrum of activities including management, operations, and administration. What differentiates this article from others is that we reflect on experiences and knowledge gained through the management of the InteropNet a very large, dynamic, heterogeneous, multi-protocol network. We hope that the experiences and knowledge gained through this effort can be incorporated into the design of future network and systems management products.

The article is organized as follows: We present an overview of the InteropNet starting with its design and ending with its deployment at NetWorld+Interop, then we outline our management strategy and, lastly, present a discussion of the lessons learned from such an undertaking.

The InteropNet

In this section we present a brief overview of the InteropNet ranging from its design and installation to its deployment at show venues. Previous articles [1, 4, 10, 11] document the InteropNet as it existed some time ago. Although still applicable, the dynamics of the InteropNet warrant periodic discussion. Pitsker [16] documents the InteropNet as it existed in 1993 while a World-Wide Web page [9] expands and provides more updated information.

Over the years, an InteropNet design criteria has evolved which stresses flexibility, interoperability, modularity, and transportability.

Flexibility is important for several reasons: First, the network design must be extremely flexible so to accommodate changing requirements and requests from users, component changes, and equipment failures. Second, flexibility is key due to the desire to incorporate the latest, emerging network technologies. Third, because the InteropNet is built almost exclusively from donated and loaned equipment, a design lacking flexibility may often fail to meet the requirements of a specific show given the available equipment.

It almost goes without saying that interoperability is an important criterion. Indeed, the original purpose of the Interop organization (and the very basis of its name) was interoperability testing. To that end, participating in the InteropNet is an important avenue for demonstrating interoperability and conformance to industry and organizational standards.

Modularity is becoming even more important as the NetWorld+ Interop show expands; There are currently seven shows worldwide per year, with each show placing differing constraints and requirements on the InteropNet. One feature of this modular design permits sections of the network to be deployed and installed independently of one another. Lastly, transportability is obviously important because the network must be quickly shipped, deployed, assembled, and disassembled throughout the world.

Logical design

The design criteria above has led to a "backbone and rib" design whereby multiple, redundant backbone networks feed rib networks. Rib networks, in turn, feed exhibitors and users. Normally, there are at least two backbone networks (currently ATM and dual-ring FDDI). Rib networks generally are Ethernet with ATM, FDDI, 100BaseT and 100VG-AnyLAN added at the larger venues. ISDN is also a planned technology for 1996 or 1997. In addition to rib networks, the backbone interconnects special purpose networks in hotels, Network Application Centers, (see below) and the Internet. For each, two separate, redundant routes are provided so that network connectivity can be preserved if failures occur.

Physical design

Groupings of equipment racks termed "peds" make up the entire InteropNet. Peds (short for "pedestals," an historic reference) are advantageous for several reasons. First, they can be transported quickly and safely because all sensitive network equipment is mounted within and protected by equipment racks. Second, the pedestals can be assembled and configured off-site before they are deployed at a particular show. Two types exist: concentrator (or "C") peds and router (or "R") peds. Concentrator peds generally serve to aggregate traffic from ribs and network segments and connect to router peds. Router peds contain at least two routers and/or switches; their primary task is to route between rib segments and the multiple, redundant backbone networks.

Network Application Centers

Dispersed throughout the exhibit halls and surrounding hotels are *Network Application Centers* (NACs) which provide banks of computers and X-terminals that allow attendees to access the InteropNet, the Internet, and the World-Wide Web. In addition, NACs often feature software and hardware from participating exhibitors and have become tremendously popular. Connecting off-site NACs with the InteropNet is accomplished using digital telephone lines, microwave transceivers, and point-to-point lasers at up to 155Mbps.

continued on next page

Managing the InteropNet *(continued)*

Network Operations Center

The *Network Operations Center* (NOC) is the heart of the InteropNet. From the NOC, a team of engineers manage and troubleshoot the InteropNet. The NOC is also where all rib and backbone networks within the InteropNet come together and connect to the Internet through multiple, redundant links. For the Atlanta 1995 show, two separate 45 Mbps links to two different providers were used.

Exhibitor connectivity

Each exhibitor is required to connect to the InteropNet, and many choose to connect at multiple points using Ethernet, Token Ring, FDDI, or ATM. Further, some vendors require special connectivity to other exhibitors as part of special demonstrations and interoperability testing. These connections, called "specials," are installed specifically for requesting exhibitors and usually require a separate cable run between booths. Requests for "specials" occur up to and throughout the show.

Building the InteropNet

Building the InteropNet is an immense effort involving hundreds of individuals. A smaller, core group of individuals (termed the "NOC Team") is responsible for the design, deployment, and management of the InteropNet. Scores of additional volunteers (*InteropNet Team Members* or "ITMs") help in this giant effort. The NOC Team is a group of highly skilled engineers with broad technical expertise, a commitment to open standards, devotion to the success of the InteropNet and the ability to lead the corps of volunteers who help with specific elements of the network. The NOC Team includes premier network engineering talent from industry, academia, government and the NetWorld+Interop staff. This team spends thousands of hours developing the network design and sleepless nights constructing a working version of the InteropNet during hot staging in NetWorld+Interop's facilities. They—with the help of ITMs—install, operate, and tear down the InteropNet at each NetWorld+Interop World Tour location.

Network and Systems Management

Managing the InteropNet is a very challenging task due to its size, its rapid deployment and installation, changing requirements, open connectivity, and its heterogeneity. Network and systems management of the InteropNet is crucial to the success of the show. Vendors invest a substantial amount of money and time to attend a trade show and have come to expect a fully-operational, production show network on which to base their marketing demonstrations. In this section, we first elaborate on the term "management" and then discuss the overriding goals and methodologies we use to manage the InteropNet.

The phrase "network management" is a somewhat loose term with widely varying definitions. While some separate the day-to-day tasks of running the network from its management, others bundle the entire spectrum of operations, administration, maintenance, and planning as network management. Further, some taxonomies separate the management of the network from management of connected systems. Our definition of network management has evolved over time to include the operations, administration, maintenance, planning, and troubleshooting of almost all facets of the InteropNet including network components and critical systems.

Management goals

The overriding goal of network management of the InteropNet is the provision of network connectivity to exhibitors and attendees. That is, first and foremost, full connectivity must be provided between every exhibitor, the Network Application Centers, and the NOC.

Secondary goals include connectivity to the outside world (via the Internet), network security and integrity, and the provisioning of a minimum level of network quality-of-service to all devices. Unfortunately, some goals may never be fully realized (e.g., network security and integrity) while others may not be feasible given current network architectures (e.g., IP and Ethernet and reliable quality-of-service guarantees for the fair allocation of network bandwidth.) Further, we differentiate between the management of the "core" InteropNet with management of exhibitor equipment.

Our management goals differentiate between exhibitor equipment (termed "non-core" equipment) and InteropNet (or "core") equipment even though all are interconnected. This differentiation is based primarily on the fact that we (the NOC team) cannot provide network and systems management for every single device (e.g., exhibitor equipment) attached to the InteropNet. The distinction is often based on who maintains administrative control. Unfortunately, failures and errors in non-core equipment can and do have a tremendous impact on the correct functioning of the InteropNet. For those cases, we must be able to identify, isolate, and recover from such problems.

Management strategy

The overall management strategy we use evolves over time as knowledge is gained and new management technologies appear. However, first and foremost, our management strategy is shaped by and is no better than the very hardware and software tools and technologies at our disposal. Indeed, we often find it necessary to augment commercially available tools with those developed expressly for use in managing the InteropNet. Nevertheless, our management strategy is centered around detecting and correcting faults before they impact network connectivity. As can be expected, this goal may not always be realized. Further, no single network and systems management technology can suffice for all management needs.

Not surprising, however, the overriding management technology used on the InteropNet is the *Simple Network Management Protocol* (SNMP) [3, 17]. Although not exclusive, SNMP is a major component of our management strategy, providing a foundation for system and network management. Virtually all equipment used in the InteropNet is SNMP manageable—even the uninterruptible power supplies (UPSs) contain SNMP agents. Despite its ubiquity, SNMP is no substitute for hand-held analyzers and administration tools like *ping, traceroute,* and *telnet.* While early shows functioned without SNMP [4], the InteropNet of today would not be manageable without it. Lastly, despite its lowly status in the SNMP community, SNMP `Trap` messages are an extremely important part of our overall management strategy, providing trigger alarms and active warnings as well as exception-based management.

Our management architecture (built to realize the management strategy articulated above) has been simultaneously developed from the top down as well as from the bottom up. It is a relatively flat hierarchy consisting of roughly four layers and mirrors our overall network design. Those four layers are: ribs, backbone and external connectivity, systems/NOC, and management operations. The various management technologies are then deployed according to how they help us manage each of the layers.

Rib management

Because exhibitors and network application centers are connected to the InteropNet through rib segments, their management and monitoring is the first component of our management architecture.

continued on next page

Managing the InteropNet *(continued)*

We wish to be able to detect and recover from hardware and software failures, configuration errors, as well as failures in any non-core equipment which impacts the rib or InteropNet. To that end, we deploy a host of equipment to remotely monitor and troubleshoot ribs. First, we utilize RMON [21, 26] probes and RMON agents for statistics collection and event generation. Second, we monitor MIB-2 [6] and private-enterprise MIB statistics for each router interface attached to each rib. Third, we often have distributed protocol analyzers attached to each rib that permit us to remotely capture and analyze rib packets. Fourth, we utilize hand-held portable and wireless tools to spot-check and troubleshoot ribs. Lastly, a special "spy" network can be used to actively monitor any rib from a physical location in the NOC.

The spy network is a series of point-to-point fiber links used in conjunction with optical/electrical switches that enable managers to place workstations, hand-held analyzers, and other management devices in the NOC onto a rib segment without the need for network-layer routing. The spy network can be thought of as providing the ability to virtually patch any NOC machine directly onto any rib.

Backbone management

Managing the InteropNet backbone is crucial to the success of the show because without its proper functioning, little network connectivity would exist. In some respects, backbone management is simpler than rib management because only core equipment is ever directly connected to it. However, an exhibitor's mis-configured router can cause routing problems on the rib it is connected to. One common problem is "black-holing" a rib. This occurs when a mis-configured router advertises that it has the "best routes" on the rib. Consequently, all exhibitor traffic on that rib goes to that router and disappears. The InteropNet insulates itself from such problems by configuring its routers to only exchange routing information with other InteropNet routers. However, these kinds of problems can appear and will affect exhibitors. We have developed a number of techniques for detecting black-holes, including the active monitoring of RIP packets.

Our major backbone management tasks include the collection of statistics, route management, and detection and recovery from hardware and link failures. Backbone and router management are so important to the InteropNet that a small, separate group with the NOC team devote their entire time to that task. Router monitoring and backbone statistics collection are performed using SNMP, MIB-2, and private-enterprise MIBs. Indeed, SNMP is ideally suited for this kind of monitoring. Route management is performed differently using a variety of techniques. First, management software periodically polls a set of dispersed machines in order to test network connectivity and reachability. This periodic polling enables us to detect link and hardware problems as well as routing protocol failures. However, this technique alone is not sufficient because the inability to reach a node may be due to a variety of problems. To perform route management, we have experimented with a specialized route verification tool that monitors OSPF [15] and RIP [8] routing protocol messages; this tool tracks the topology as constructed via routing protocols and compares it against a pre-programmed topology. When topology changes occur, the route verification tool sends event messages and raises alarms. To detect hardware and link failures, we rely on hand-held analyzers as well as SNMP `Trap` messages.

Systems/NOC management

Systems Management is becoming more important as the correct functioning of networks increasingly relies on the health of key systems. For example, in many networks today the operation of key services like DNS [14], NIS, and Web [2] servers are crucial to the health of a network and the services it provides. Not surprising, the correct functioning of many NOC systems is crucial to the overall operation of the InteropNet. To better manage critical systems, we use SNMP agents supporting the Host Resources [7] and Systems Management [12, 13] MIBs. These agents allow us to monitor critical processes, track system resources, and monitor the overall health of systems so that we can detect and prevent systems-related problems before they occur.

Management operations

The last layer in our management architecture is that composed primarily of SNMP management software and management data manipulation tools and scripts. Our SNMP management software is composed of enterprise management platforms, element managers, special purpose software, and SNMP browsers [19].

We use enterprise management software quite extensively for statistics collection, basic polling and reachability testing, and the graphical depiction of the InteropNet. Although we distribute enterprise management across the InteropNet, manager-to-manager communication (at present) is almost nil for a variety of reasons. First, manager-to-manager communications is still proprietary, although some SNMPv2 work [5] has addressed this problem. Second, we desire management autonomy for increased robustness. Third, we are not entirely convinced that manager-to-manager communications adequately addresses many of our network and systems management problems.

We make heavy use of element managers as well as vendor-specific and special purpose management software. Element managers provide increased management of classes of devices (e.g., hubs or routers) while vendor-specific management software is often used because general purpose and element software often lack sufficient semantic understanding of private-enterprise MIBs. Unfortunately, the insistence by many companies to use private MIBs for functions available in public MIBs and other issues render vendor-specific element managers a requirement.

We make heavy use of specialized management software to "fill the gaps" left between general purpose, element management, and vendor-specific software. One example is our development and use of a *Trap-exploder*. The Trap-exploder [22] is a software tool that allows us to receive SNMP `Trap` messages on a single system, log them to a file, and forward them to other management stations, element managers, and vendor-specific software. The Trap-exploder greatly reduces configuration overhead because we can designate a single InteropNet machine as a recipient of all SNMP `Trap` messages. We also use customized software and scripts to count the number of devices connected to the InteropNet. Lastly, one of the most commonly used SNMP management tools is a graphical MIB browser which diagrams MIB modules in a point-and-click interface. This type of tool provides a common interface for accessing any standard or private-enterprise MIBs without requiring specialized vendor-specific management software, provided the MIBs are available and can be compiled using standard MIB compilers. When troubleshooting, we often do not have the time to invest in learning vendor-specific management software.

continued on next page

Managing the InteropNet *(continued)*

Reflections and lessons learned

The development of a management strategy and architecture as well as its use on the InteropNet has provided a tremendous opportunity to learn the art and science of network and systems management in addition to providing us the opportunity to thoroughly test management software, practices, and accompanying frameworks. In this section, we articulate some of the important lessons we have learned over the course of the past few years. We hope that some of these lessons can be incorporated into future product design and implementation. Our observations range from product and framework deficiencies to more general industry observations.

Framework deficiencies

Our experiences have highlighted what we feel our deficiencies in the Internet Management Framework (SNMP) when applied to a large, dynamic, and heterogeneous networks such as the InteropNet. One problem we have frequently encountered revolves around the looseness of SNMP MIB specifications. This looseness, and differences of interpretation, has led to interoperability problems between management and agent implementations from different vendors. For example, we find that management software designed to work with a standard MIB often is incompatible with another vendor's implementation of that MIB. This incompatibility prevents us from using a single management application; consequently we must often install numerous, overlapping management applications, which increases our configuration overhead and resource usage. Another problem involves the lack of semantic expressiveness of SNMP MIB specifications. The current standard SNMP MIB format [18] does not permit the expression of causality between and among managed objects. For example, MIB specifications should permit the linking of managed objects and managed object values to other managed objects as well as `Trap` messages. Lastly, the lack of security within the current framework limits some of our SNMP-based management to monitoring only. However, to increase the security of InteropNet devices, we install a special Ethernet segment (called "access ether") over which most management traffic is routed. This segment is not accessible by exhibitors or attendees.

Implementation flaws

Our experiences have also highlighted what we feel are product implementation deficiencies when applied to networks larger than a single, small, isolated LAN environment. Because software bugs are an unfortunate fact of life and are fixable, we will only focus on design issues here. One major problem we encounter is the lack of flexible element management software; this deficiency has led to the balkanization of many management tasks. Hub management, for example, is very difficult. When a hub-specific `Trap` is received, we must first determine which vendor manufactured the hub and then navigate the appropriate vendor-specific hub management software. We currently cannot use vendor B's hub management software with vendor A and vice versa. Poor integration of element and management station software only adds to this problem. It is clearly in the best interest of both the industry and individual vendors to solve these problems and enable interaction and interoperability between element managers, and integration with enterprise management systems. The lack of integration is an embarrassment to the industry and clearly a concern for all network managers.

Interoperability and robustness concerns

Another category of implementation deficiencies centers around interoperability concerns. One huge problem we encounter at every show is MIB compilation problems.

Vendors continue to ship pre-Concise-MIB specifications as well as MIBs so filled with syntax errors that they are unusable. While it is tempting to categorize this class of problems as implementation bugs, we feel that it occurs with such regularity as to constitute either a design flaw or an intentional oversight. It seems vendors generally never use, attempt to compile, or test their own MIBs with other management software. We regularly encounter problems with management software reliance on the functioning of systems-related services like DNS, NIS, and NFS [20]. When management software unnecessarily relies on the correct operation of system services, and those system services become unavailable, management software ceases to be functional and only aggravates the problem. For example, many management station implementations rely on DNS and NIS address lookup despite the fact that we have configured network layer addresses into their management databases. When DNS services are unavailable, the management station software becomes unusable.

Software configuration

Management software is often so complex and difficult to operate that mis-configuration itself can lead to network problems. One more humorous example involves the development of the Trap-exploder software. We were seeing exceedingly large numbers of **Traps** early in a pre-show environment, many coming from core equipment. We struggled with the problems in an effort to correct what appeared to be catastrophic meltdown of the entire network. It turns out that we had configured the Trap-exploder machine as a receiver of **Traps** from the Trap-exploder, thus creating a **Trap** delivery loop. When this occurred, the Trap-exploder would forward received **Trap** messages back to itself, resulting in almost instant implosion of the underlying machine! Another common problem involves the immense configuration overhead necessary to use most commercial management stations. Although this problem may be more specific to the fast-paced, short-lived environment like NetWorld+Interop, the daunting task of configuring management station software limits its usefulness.

Lastly, we have observed a general trend towards poor "factory" configuration of most management and agent software. For example, many hubs are configured by default to send **Trap** messages at such a high rate (say one per minute or even more frequently) so as to inundate a high-powered workstation and render all **Trap**-based management useless. Further, most devices are configured to send authentication failure **Traps** by default and many do not support the ability to change this behavior via an SNMP **set** request. In addition, private-enterprise MIBs are often written such that read-write community strings are contained within tables that can be queried using read-only permissions. Consequently, any browser can discover read-write community strings and compromise any management security that may exist.

An industry trend

We have also noticed a general industry trend forming which we term "open proprietary computing" or "OPC." The oxymoron is intentional and is used to describe a practice becoming increasingly common. For example, many vendors encode their private-enterprise MIBs within their management software, but do not distribute the MIB specification to users of their products. This situation prevents network managers from picking and choosing the best management software independent of network hardware. This practice, which greatly disturbs us, is tantamount to product tying as well as the closing of an open standard.

continued on next page

Managing the InteropNet *(continued)*

Another example of open proprietary computing can be found in vendor's minimalist implementations of standard MIBs, which they then augment by full-featured private-enterprise MIB implementations tied to their own management software. This implementation strategy is similar to bait-and-switch selling. These trends are accelerating and appear to be aimed at creating a new paradigm for "golden handcuffs."

Positive trends

We also have noticed many positive developments in the network and systems management arena. SNMP's successful deployment and near ubiquity have greatly enabled the remote monitoring of network equipment and systems. New private-enterprise and standard MIBs are emerging that will greatly enhance our management ability. The SNMPv2 process is addressing and improving the expressiveness of MIB specifications, addressing some of the root causes of interoperability problems, as well as addressing scalability issues. Lastly, new management software appears to be improving in several key areas: it is providing increased integration as well as the ability to be "programmed" with or learn the knowledge of its operators.

Conclusion

Building and installing the InteropNet presents many challenges due to its size, its dynamism, and its heterogeneity. Managing such a network presents tremendous problems, but also provides for unique insights into the strengths and weaknesses of current network management practices and products. We have articulated the network and systems management goals of the InteropNet NOC team as well as the basic architecture we use to fulfill them. We then discussed a few weaknesses of the components that make up our management architecture and hope that our experiences will guide future design and development. Although we tended to focus on many of the problems, many things do work and do work very well. As Dave Clark has said: "keep the faith!"

Acknowledgements

A great many people participate in building, designing, deploying and managing the InteropNet. So many, in fact, that we cannot individually acknowledge them. All contribute greatly; without all their help, the InteropNet would not exist.

References

[1] P. Almquist, "The INTEROP 88 network—behind the scenes," *ConneXions,* Volume 3, No. 2, February 1989.

[2] Tim Berners-Lee, "Hypertext transfer protocol (HTTP)," Internet Draft, November 1993.

[3] J. D. Case, M. Fedor, M. L. Schoffstall, and C. Davin, "Simple network Management Protocol," RFC 1157, May 1990.

[4] B. Chapman, "Building & managing the INTEROP 91 Fall Shownet," *ConneXions,* Volume 6, No. 6, June 1992.

[5] Jeffrey D. Case, Keith McCloghrie, Marshall T. Rose, and Steven L. Waldbusser, "Manager-to-manager Management Information Base," RFC 1451, April 1993.

[6] K. McCloghrie, M. Rose, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II," RFC 1213, March 1991.

[7] P. Grillo and S. Waldbusser, "Host Resources MIB," RFC 1514, September 1993.

[8] C. Hedrick, "Routing Information Protocol," RFC 1058, June 1988.

[9] "Pocket guide to the InteropNet," NetWorld+Interop 1995. `http://www.interop.net/interopnet/brochure.html`

[10] S. Knowles, "The INTEROP 89 network: from one of its builders," *ConneXions,* Volume 4, No. 2, February 1990.

[11] S. Knowles, "Building the INTEROP 90 Show Network," *ConneXions,* Volume 5, No. 9, September 1991.

[12] B. Krupczak, "Unix systems management via SNMP," In Proceedings of the IFIP TC6/WG6.6 Third International Symposium on Integrated Network Management, April 1993.

[13] B. Krupczak, "Systems Management and the Internet Management Framework," *ConneXions,* Volume 9, No. 8, August 1995.

[14] P. Mockapetris, "Domain names—implementation and specification," RFC 1035, November 1987.

[15] J. Moy, "OSPF Version 2," RFC 1583, March 1994.

[16] Bo Pitsker, "Insights into the INTEROPnet," *ConneXions,* Volume 7, No. 3, March 1993.

[17] M. T. Rose and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based internets," RFC 1155, May 1990.

[18] M. T. Rose, "Concise MIB Definitions," RFC 1212, March 1991.

[19] Marshall T. Rose, "Network management: Status and challenges," *ConneXions,* Volume 7, No. 6, June 1993.

[20] Sun Microsystems, Inc., "NFS: Network File System protocol specification," RFC 1094, March 1989.

[21] Steve Waldbusser, "Remote Network Monitoring Management Information Base," RFC 1271, November 1991.

[22] For a copy of the Trap-exploder, send e-mail to: `info@empiretech.com.`

[23] Hares, S., "Lessons Learned for OSI at INTEROP 91 Fall," *ConneXions,* Volume 6, No. 3, March 1992.

[24] Jacobsen, O., "INTEROP 88 Conference Report," *ConneXions,* Volume 2, No. 11, November 1988.

[25] Dern, D., "Highlights from INTEROP 89," *ConneXions,* Volume 3, No. 11, November 1989.

[26] Stallings, W., "RMON2: The Next Generation of Remote Network Monitoring," *ConneXions,* Volume 10, No. 5, May 1996.

BOBBY KRUPCZAK is Chief Scientist at Empire Technologies, Inc. where he designs and implements intelligent network and systems management agents and managers. He regularly participates on the InteropNet NOC team where has gained valuable insight into the art of network and systems management of large, heterogeneous networks. He can be reached at `rdk@empiretech.com`

STEVE HULTQUIST is President of Worldwide Solutions, Inc. in Boulder, Colorado, which specializes in helping organizations design and deploy technology-based business solutions. Steve has an extensive background in information technology, from application development to outsourcing, and he has a special interest in designing, building, and deploying networks. He regularly participates on the InteropNet NOC team. He can be reached at `ssh@wwsi.com`

The User-based Security Model for SNMPv2

by Glenn W. Waters, Bell-Northern Research Ltd.

Introduction

The *User-based Security Model* for SNMPv2 (USEC) provides an administrative framework through which multiple levels of security for SNMPv2 protocol interactions can be defined. It achieves this with a minimum of overhead on the management and agent entities.

This article gives an overview of the concepts and security features of the USEC model. The USEC model also defines mechanisms to handle proxy agents that is not covered in this article.

The *user,* a well known paradigm for computer users, is the basis of all protocol interactions in the USEC model. Conceptually, the user model is very analogous to the user-id/password that is used to login to a computer. The user, identified by a *username* (the "user-id") is used to identify who is accessing information at an agent. A key (the "password") is used to ensure that the user is authentic. Optionally, a second key may be used to ensure privacy. The user-id/password model also grants a user-id a set of privileges. Similarly, the USEC model associates a set of access rights with a user.

Implicit in the user-id/password model is that some entity (machine or human) has knowledge of the user-id and password that may be used to login to a particular computer. The USEC model maintains that paradigm in that the shared knowledge of a user must be known to both the entity that wishes to access an SNMPv2 agent and to the agent itself.

Historical Perspective

Just as several years worth of work on a new security and administrative framework for SNMP version 1 (SNMPv1) were published (RFCs 1351–1353), a new effort to was started on the design of version 2 of SNMP (SNMPv2). Rather than having two transitions, one from SNMPv1 without security to SNMPv1 with security and a second transition to SNMPv2, the two efforts were combined under the moniker of SNMPv2. The first fruits of the combined effort was the publishing of 12 RFCs (RFCs 1441–1452) by the SNMP working group. These RFCs, containing over 400 pages of text, were considerably more complex than the original SNMP. However, a number of useful advances were defined in the new documents, most notably:

- The oral tradition of the original SNMP was documented;
- A number of enhancements to the language for defining managed objects; and,
- Performance improvements were made to the protocol.

The SNMPv2 RFCs also contained much of the security and administrative framework that had been defined in the prior years. This framework was based on the notion of a *party* and a *context.* The party and context contained information about:

- *Keys,* used to authenticate and encrypt messages;
- *Clocks,* used to detect late or retransmitted messages (caused inadvertently or maliciously);
- *Access rights,* used to specify the valid protocol operations and on which managed objects those operations were valid; and,
- *Transport,* which specified what network service (e.g., UDP) and network address should be used to receive messages.

In theory, the party model was very elegant and well-defined, and, although implementation was much more difficult than SNMPv1, it was still very tractable. The problem arose in deployment of a party-based administrative model. For every agent that a management station wished to communicate with, two parties had to be defined at the management station and an almost identical two at the agent. As the number of agents (i.e., managed nodes) increased, the number of parties increased, resulting in a large administrative problem.

What further compounded the problem was that if a second management station wished to communicate with the same set of agents as the first management station, new sets of parties needed to be defined at the management station and at each of the agents. Not only did this further increase the number of parties to administer, this was a very unreasonable requirement for the end-user since deploying a new management station meant that the new parties had to be pre-configured before the station could become operational. If a new management station was required to be deployed during a time of network stress (i.e., the user was actually trying to do some network management), there was very little chance that the parties could be configured.

Ultimately, these limitations led to the demise of the security and administrative model that was defined in SNMPv2. Last minute attempts to define a new solution to the problem met in deadlock within the SNMPv2 working group. As an attempt to salvage the useful features in SNMPv2, the party-based approach to security and administration was replaced in the SNMPv2 specification with the SNMPv1 community-based model. This model, termed the *Community-based SNMPv2* (SNMPv2C), provides exactly the same security and administrative features as is defined in SNMPv1. Namely, a plaintext in the clear community string is the basis for all security and administrative features.

Not being satisfied without a proper security and administrative model, a simplified security model called the User-based Security model was defined. This is the subject of this article. The USEC model described here is defined in RFC 1910. [3]

Goals and Constraints

The specific goals of the USEC model with respect to security, are:

- To provide verification that each received SNMPv2 message has not been modified during its transmission such that an unauthorized management operation might result;
- To provide verification of the identity on whose behalf the SNMPv2 message claims to have been generated;
- To provide verification of timeliness of received messages; and,
- Optionally, to ensure that the contents of SNMPv2 messages are protected from disclosure from unauthorized sources.

The USEC model specifically does not attempt to protect against denial of service attacks and against traffic analysis.

The goals and non-goals defined here are, in practice, the same as those defined for the now-historical SNMPv2 party model specified in RFC 1446. [4]

continued on next page

The User-based Security Model *(continued)*

Definition of a User

A user has the following attributes:

- *userName,* an octet string that represents the identity of the user;
- *authProtocol,* an indication of whether messages sent on behalf of this user can be authenticated, and if so, the type of authentication protocol which is used;
- *authPrivateKey,* if messages sent on the behalf of this user can be authenticated, the private authentication key for use with the authentication protocol;
- *privProtocol,* an indication of whether message sent on behalf of this user can be protected from disclosure, and if so, the type of privacy protocol which is used; and,
- *privPrivateKey,* if messages sent on behalf of this user can be protected from disclosure, the private privacy key for use with the privacy protocol.

Those attributes must be known to both the sender and the receiver of a communication.

The USEC model defines the use of MD5 as the authentication protocol and the use of DES as the privacy protocol. The keys associated with both the MD5 and DES protocols are 128-bit values. To provide an interface that is more user-friendly than 128-bit keys, USEC model specifies the use of a password-to-key algorithm as was originally defined by Steve Waldbusser as part of the SNMPv2 party model.

Other Definitions

Agents that support the USEC model must maintain three objects:

- *agentID,* which is a 12 octet identifier that is unique among all agents in an administrative domain;
- *agentBoots,* which is a count of the number of times the agent has rebooted/re-initialized since the *agentID* was last configured (each time the agent is re-initialized the value of *agentBoots* must be incremented by one); and,
- *agentTime,* which is the number of seconds since *agentBoots* was last incremented.

Quality of Service (*qoS*) is defined as the level of security that is afforded a particular message. The defined *qoS* levels are:

- No authentication (*noAuth*) and no privacy *(noPriv)*;
- With authentication *(auth)* and no privacy (*noPriv*); and,
- With authentication (*auth*) and with privacy (*priv*).

A message's *qoS* is included within the message's header, and *qoS* is also used to indicate messages that may result in a report PDU being generated.

Time Window is a value that specifies the window of time in which an authenticated message generated on behalf of a user is valid. The same value of the Time Window, 150 seconds, is used for all users.

Local Configuration Datastore (LCD) is a locally defined (conceptual) datastore that holds a set of information about (locally known) SNMPv2 users and other associated information (e.g., access control). Each SNMPv2 entity maintains an LCD.

An LCD may potentially be required to hold information about multiple SNMPv2 agent entities (e.g., in a manager that communicates with multiple SNMPv2 agent entities), and as such the *agentID* should be used to distinguish the information associated with a particular agent entity in the LCD.

Message Format

The format of a message using the USEC model is the same as the SNMPv1 message specified in RFC 1157 [5], except that:

- The version number is changed to 2; and,
- The data component contains either a PDU or an `OCTET STRING` containing an encrypted PDU.

In addition, the SNMPv1 community string component in the message, termed the *parameters* component in the USEC model, contains a set of administrative information for the message.

An SNMPv2 message is an ASN.1 value with the following syntax:

```
Message ::=
    SEQUENCE {
     version
          INTEGER { v2u(2) },

     parameters
          OCTET STRING,

     data
          CHOICE {
               plaintext PDUs,
               encrypted OCTET STRING
          }
    }
```

The first octet in the parameters component is the *model,* where the value 1 refers to that USEC model. In that case, the parameters field contains several values encoded in network-byte order:

- *qoS,* the message's quality-of-service;
- *agentID,* the unique identifier for the agent;
- *agentBoots* / *agentTime,* the message's timestamp;
- *maxSize,* the maximum message size which the sender of this message can receive using the same transport domain as used for this message;
- *userLen* / *userName,* the user on whose behalf this message is sent;
- *authLen* / *authDigest,* the authentication digest; and,
- *contextSelector,* the context selector, which in combination with *agentID* identifies the SNMPv2 context containing the management information referenced by the SNMPv2 message.

The *plaintext* field contains an SNMPv2 PDU as defined in RFC 1905, [8] whilst the *encrypted* field contains the encrypted form of an SNMPv2 PDU.

Contexts and Context Selectors

An SNMPv2 context is a collection of management information accessible by an SNMPv2 agent. An item of management information may exist in more than one context. An SNMPv2 agent potentially has access to many contexts.

continued on next page

The User-based Security Model *(continued)*

Each SNMPv2 message contains a context selector which unambiguously identifies an SNMPv2 context accessible by the SNMPv2 agent whose *agentID* is contained in the message.

A context is termed a *local* SNMPv2 context if it is realized by an SNMPv2 entity that uses locally-defined mechanisms to access the management information identified by the SNMPv2 context.

The term *remote* SNMPv2 context is used at an SNMPv2 manager to indicate an SNMPv2 context which is not realized by the local SNMPv2 entity (i.e., the local SNMPv2 entity uses neither locally-defined mechanisms, nor acts as a proxy SNMPv2 agent to access the management information identified by the SNMPv2 context). The USEC model also defines proxy SNMPv2 contexts.

The combination of an *agentID* value and a context selector provides for a unique identification of a context within an administrative domain.

Error Reporting

While processing a message, an SNMPv2 agent entity may determine that the contents of the message header is unacceptable according to one of the requirements of the USEC model. Rather than just discarding the received message, and force the management entity to await a timeout period to detect that an error condition has occurred, the agent may generate a message containing SNMPv2's new **`report`** PDU.

When the agent entity detects an error (e.g., the received message has a bad authentication digest) and the *qoS* indicates that a **`report`** may be generated, then after incrementing the appropriate error statistic, a **`report`** PDU message is generated. The **`report`** is generated with the same user and context as the received message and is sent to the same transport address as the received message. All error reports, except those generated due to a not-in-time-window error condition, are unauthenticated (i.e., *qoS* is *noAuth / noPriv*). To allow a management entity to authentically synchronize its time with the agent's time, those error reports generated due to a not-in-time-window error condition are authenticated (i.e., *qoS* is *Auth / noPriv*).

Upon receiving a **`report`** PDU a management entity may perform any error recovery actions that are appropriate, such as performing automatic error recovery (i.e., clock synchronization) or notifying the management station operator of the error condition.

The report flag in the *qoS* may only be set if the message contains a **`get`**, **`get-next`**, **`get-bulk`**, or **`set`** operation. The report flag should never be set for a message that contains a **`response`**, **`inform`**, **`trap`**, or **`report`** operation. Furthermore, a **`report`** PDU is never sent by an SNMPv2 entity acting in a manager role.

Message Authenticity

To ensure message integrity and to verify the identity of the user on whose behalf a message is sent, the MD5 message digest algorithm described in RFC 1321 is used as follows:

- The user's 128-bit secret key, *authPrivateKey,* known to both the sender of the message and the receiver of the message is inserted into and appended to the SNMPv2 message.
- Using the MD5 message digest algorithm, a 128-bit digest (checksum) is computed over the entire message and appended secret. The computed digest is inserted into the message, replacing the secret value, and the resulting message is transmitted to the recipient.

- The recipient of the message replaces the 128-bit digest with the secret value, saving the digest for later use, and appends the secret value to the received message. Using the MD5 message digest algorithm, a 128-bit digest is computed over the entire message and the appended secret. The computed digest value is compared to the received digest value and if they are equal then the message's integrity is intact and its identity of origin is deemed to be authentic.

Any message that is not authentic is discarded, possibly causing a **report** PDU message to be generated.

In comparison, the party model did not append the 128-bit key to the message before the digest was computed. This usage of MD5 is called *Keyed-MD5,* and, according to security experts, it cryptographically strengthens the algorithm. Furthermore, a user's keys are *localized* for each agent. This gives superior security properties, as outlined in [11].

Timeliness of Delivery

Each SNMPv2 agent is the authoritative source of two time values, *agentBoots* and *agentTime,* which taken together provide an indication of time at that agent. When a manager wishes to authentically communicate with the agent, it must include its notion of both of these values in the message. On receipt of a message at the agent, the *agentBoots* and *agentTime* values are checked to ensure that they are within an acceptable time window (150 seconds) of the agent's current time.

When an agent generates a message, its current values of *agentBoots* and *agentTime* are always included in the message. When an authentic message is received by the manager, the *agentBoots* and *agentTime* values in the message are checked to ensure that they are within an acceptable time window of the manager's local values for the agent's time.

Any message that is not within the acceptable time window is discarded, possibly causing a **report** PDU message to be generated.

Replay Protection

As discussed previously, each SNMPv2 agent must maintain three objects, *agentID, agentBoots,* and *agentTime.*

The *agentID* value is used to protect against attacks in which a message from a manager is replayed to different agent and/or messages from one agent are replayed as if from a different agent. Since *agentID* is unique within an administrative domain and the *agentID* is included in the portion of a message that is authenticated, the same message from/to different agents will contain a different MD5 digest, even if the same *authPrivateKey* is used. This prevents messages from being cross-played.

To protect against replay, authentic messages are checked to ensure that they are timely. A management entity will accept a received message as timely:

- If the received value of *agentBoots* is greater than the local notion of *agentBoots;* or
- If the received value of *agentBoots* is equal to the local notion of *agentBoots* and the received *agentTime* is not more than 150 seconds less than the local notion of *agentTime.*

continued on next page

The User-based Security Model *(continued)*

Simply stated, the timely message is one that contains an indication of time that may be greater than the local notion of the agent's time and is no more than 150 seconds older than the local notion of the agent's time.

An agent entity will accept a message as timely if the message contains a value of *agentBoots* equal to the agent's current value of *agentBoots* and the message contains a value of *agentTime* that is within 150 seconds of the agent's current *agentTime*. The agent checks that the received time is strictly within the 150 second window to protect against a manager irresponsibly using an indication of time that is at some arbitrary point in the future, thus allowing captured messages to be replayed until they are no longer timely.

In order for the mechanisms described here to reliably protect against reply attacks, the indication of time at an agent must be ever increasing once the agent is installed and running. Through the use of *agentBoots,* a non-volatile clock which ticks at all times (even when the agent is powered down) is not required at the agent. The *agentBoots* value is simply incremented when the agent re-initializes and the agent's indication of time has then advanced, thus providing protection against replay attacks.

Both *agentID* and *agentBoots* must be stored in non-volatile storage. If the agent cannot determine the values of *agentID* or *agentBoots* then the value of *agentBoots* should be set to its maximal value of 4294967295.

When an agent is first installed, it sets its local values of *agentBoots* and *agentTime* to zero. If *agentTime* ever reaches its maximum value (2147483647) then *agentBoots* is incremented, as if the agent had rebooted, and *agentTime* is reset to zero and starts incrementing again. If *agentBoots* reaches it maximum value (4294967295) manual intervention is required and the agent must be physically visited and reconfigured, either with a new *agentID* value, or with new secret values for all users known to that agent. Note that it would take 136 years of the agent rebooting once a second for this condition to ever arise!

Privacy

To ensure that messages are protected from disclosure from unauthorized sources, the USEC model employs the *Data Encryption Standard* (DES) in the *Cipher Block Chaining* (CBC) mode of operation. A 128-bit privacy key, known by both the sender and receiver of a message, is used to encrypt the PDU portion of the message prior to sending. Upon receiving an encrypted message the same privacy key is required to successfully decrypt the PDU portion of the message.

Time Synchronization

Time synchronization is required by a management entity in order to proceed with authentic communications with an agent entity (including being able to check the authenticity of a received `trap`). A management entity has achieved time synchronization with an agent entity when the management entity has obtained local values of *agentBoots* and *agentTime* from the agent that are within the agent's time window. In addition to keeping a local version of *agentBoots* and *agentTime,* a manager must also keep one other local variable, *latestReceivedAgentTime.* This value records the highest value of *agentTime* that was received by the manager from the agent and is used to eliminate the possibility of replaying messages that would prevent the manager's notion of the *agentTime* from advancing.

In order for a manager to become synchronized with an agent, the manager should set its local values of the agent's clocks (*agentBoots, agentTime,* and *latestReceivedAgentTime*) to zero. A subsequent authenticated message sent to the agent will cause an authentic error report to be returned to the manager. The error report, indicating that the manager sent a message that was not in the agent's time window, contains authentic values of the agent's clocks that may be used to update the manager's local notion of the agent's clocks. The manager may then continue with timely authentic communications, and, in particular, may re-send the authentic message that caused time synchronization to occur.

The manager and agent each have a clock that advance independently. For a manager to remain synchronized with the agent, its notion of the agent's time must remain current. If the manager's clock does not advance at the same rate as the agent's clock, then over time the manager's notion of the agent's time could vary enough that time synchronization will be lost. To prevent this condition, the manager's notion of the agent's time is updated (i.e., synchronized) whenever a message is received that is authentic, timely, and more recent than any other message received from that agent. Thus, a manager that maintains communication with an agent should never lose time synchronization with that agent.

Discovery

In order to communicate with an SNMPv2 agent that supports the USEC model, a management entity must know the value of the agent's *agentID* value. The *agentID* may be learned by sending a *noAuth/noPriv* retrieval communication to the agent with the *agentID* set to all zeros (binary). Since the context, identified by the combination of the *agentID* and *contextSelector,* is invalid due to an all-zero *agentID,* the agent's response to this message will be a unknown-context error `report` PDU that contains the agent's *agentID* value in the parameters field. If authentic communications are required, then the time synchronization procedure should then be used.

Access Policy

For a particular SNMPv2 context to which a user has access using a particular *qoS,* that user's access rights are given by a list of authorized operations (e.g., `get`, `set`, etc), and for a local context, a read-view and a write-view. The read-view is the set of managed object instances that may be accessed by the user during a retrieval or notification operation. The write-view is the *set* of managed object instances that may be accessed by the user when performing a `set` operation.

USEC MIB module

The USEC model defines a MIB module that contains objects for basic agent instrumentation (e.g., *agentID, agentBoots*, and so on) and for USEC statistics.

Informs, Managers, and Agents

One of the new features in SNMPv2 is the `inform` PDU, which is used to transmit management information from one "application" to another. Each of these applications act in a manager role for the purposes of sending and receiving the `inform`; however, the sending application must have access to information in a MIB view of an entity acting in an agent role. That implies that an application that sends an `inform`:

- Is a *dual-role* entity, namely, it acts as both an agent entity and a management entity;
- Must have access to the agent's MIB view; and,
- Must have access to parts of the agent's instrumentation, specifically, to the agent's authoritative time indicators.

continued on next page

The User-based Security Model *(continued)*

The concept here builds upon the fact that every (network) component should have an agent which holds its management instrumentation, and this is just as true for management applications as for any other network component. Such management instrumentation is, of course, accessible through an agent. Thus, every application should have an agent which holds its management instrumentation, and therefore every application has an associated agent with a MIB view which is used for the purpose of that application sending an **`inform`**.

With this definition, some may ask why not consider an **`inform`** as an "acknowledged trap." The answer is related to two of the age-old SGMP/SNMP philosophies:

- Keeping the "cost-of-entry" requirements on an agent to a minimum so that even the simplest of devices can afford to implement an agent; and,
- Controlling the amount of SNMP traffic generated in a network according to the needs of the management applications, i.e., only a subset of the devices in the network will generate unsolicited SNMP messages upon the occurrence of some network error. (Note that it is this philosophy which has always been the impetus behind SNMP's paradigm of *trap-directed polling.*)

It is consistent with these philosophies for certain high-end devices, which have previously been considered to be agents (e.g., RMON probes or routers), to now be considered to be dual-role entities capable of sending **`informs.`** What would be problematic is if *every* device in the network were considered to be a dual-role entity.

The USEC model requires that an application that wishes to authenticate received informs must synchronize its time with the sending application's associated agent. (This requirement is the same as is required to authenticate received traps.)

Past, Present, and Future

Considerable work has been done on USEC since its initial publication in Spring of last year. In particular, while the security of USEC has been strengthened, (e.g., through the use of localized-keys and keyed-MD5), many of the procedures have been streamlined (e.g., clock synchronization).

As of this writing there are five independently-developed and interoperable implementations of the USEC model, two commercial and three openly-available. Given the straight-forward design of USEC, it is anticipated that other implementations will be available in 2Q96.

The core of the USEC model is quite stable; however, development has started on a remote configuration MIB module for USEC. Rather than using the design team approach enjoyed by the USEC model, an open mailing list (**`usec-mib-request@fv.com`**) is now established for this purpose.

[Ed. An earlier version of this article appeared in *The Simple Times,* Volume 4, No. 1, January 1996. Used with permission.]

References

[1] Rose, M., et. at., "The USEC Resource Page," Available from: **`http://www.simple-times.org/pub/simple-times/usec/`**

[2] McCloghrie, K., Editor, "An Administrative Infrastructure for SNMPv2," RFC 1909, February 1996.

[3] Waters, G., Editor, "User-based Security Model for SNMPv2," RFC 1910, February 1996.

[4] Galvin, J., McCloghrie, K., "Security Protocols for version 2 of the Simple Network Management Protocol (SNMPv2)," RFC 1446, April 1993.

[5] Case, J., Fedor, M., Schoffstall, M., Davin, J., "Simple Network Management Protocol," RFC 1157, May 1990.

[6] Rivest, R., "The MD5 Message-Digest Algorithm," RFC 1321, April 1992.

[7] The SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., & Waldbusser, S., "Introduction to Community-based SNMPv2," RFC 1901, January 1996.

[8] The SNMPv2 Working Group, Case, J., McCloghrie, K., Rose, M., & Waldbusser, S., "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)," RFC 1905, January 1996.

[9] "Data Encryption Standard," National Institute of Standards and Technology. Federal Information Processing Standard (FIPS) Publication 46-1. Supersedes FIPS Publication 46, (January, 1977; reaffirmed January, 1988).

[10] Krawczyk, H., "Keyed-MD5 for Message Authentication," Internet Draft, IBM, June 1995.

[11] Uri Blumenthal, N. C. Hien, Bert Wijnen, "Optimizing Key Distribution," *The Simple Times,* Volume 4, No. 1, January 1996.

[12] Rose, M. T., "Network Management: Status and Challenges," *ConneXions,* Volume 7, No. 6, June 1993.

[13] Rose, M. T., *The Simple Book: An Introduction to Networking Management,* Revised Second Edition, Prentice-Hall, ISBN 0-13-451659-1, 1996.

[14] *ConneXions,* Two *Special Issues* on Network Management and Network Security, Volume 3, No. 3, March 1989 and Volume 4, No. 8, August 1990.

[15] Case, J. D., Davin, J. R., Fedor, M. S., & Schoffstall, M. L., "Network Management and the Design of SNMP," *ConneXions,* Volume 3, No. 3, March, 1989.

[16] Rose, M. T., "Network Management is Simple: You just need the 'Right' Framework." In *Integrated Network Management, II,* Iyengar Krishan and Wolfgang Zimmer, editors, pages 9–25, North Holland, April 1991.

[17] Stallings, W., "Cryptographic Algorithms, Part I: Conventional Cryptography," *ConneXions,* Volume 8, No. 9, September 1994.

[18] Stallings, W., "Cryptographic Algorithms, Part II: Public-Key Encryption and Secure Hash Functions," *ConneXions,* Volume 8, No. 10, October 1994.

[19] Kaliski, B., "An Overview of Public-Key Cryptography Standards," *ConneXions,* Volume 6, No. 5, May 1992.

GLENN W. WATERS is a senior designer at Bell-Northern Research's Ottawa Labs. He has been involved in a number network management projects since 1989 including designing and implementing a distributed, scalable network management system that is used to manage a very large internal corporate TCP/IP network. He is a member of the Internet Engineering Task Force and Editor of the User-based Security Model for SNMPv2 RFC. Glenn received his Bachelor of Mathematics from the University of Waterloo in 1983. In a life before his child arrived, he used to enjoy hiking and cross-country skiing. E-mail: `gwaters@bnr.ca`.

An Introduction to SNMPv2*

by David Partain, SNMP Research International

Introduction

This article provides a brief introduction to *SNMPv2** (pronounced "S-N-M-P-vee-two-star"), including its history, and a technical overview. The article concludes with a report on implementation status, recent interoperability testing and demonstrations, and a report on a call for restarting the open IETF standardization process leading toward the next generation of SNMP specifications ("SNMPng").

Background

The *Internet Standard Management Framework,* based on the *Simple Network Management Protocol* [1], was first introduced in 1988. This initial version (SNMPv1) was widely implemented, deployed, and used, thereby becoming a key enabling technology for the subsequent burgeoning of internet technology and the Internet. Consequently, SNMPv1 became both an open IETF standard and a *de facto* industry standard resulting from widespread market acceptance. A broad range of standard MIB documents were complemented by a vast array of vendor-specific MIB documents which collectively extended the scope of SNMP in many directions, including network management, system management, application management, manager-to-manager communication, and proxy management of legacy systems.

However, SNMPv1 suffered from multiple real and perceived weaknesses which eventually led to efforts to define SNMPv2. The SNMPv2 design process attempted to identify and address these problems.

One of the most important weaknesses of SNMPv1 is the lack of adequate mechanisms for securing the management function, including authentication and privacy, along with a suitable administrative framework defining authorization and access control. SNMPv1 also lacked a standard mechanism for remote administration of the management function. For example, it is not possible to configure SNMPv1 agents to accept queries and commands from particular management stations and to send `traps` to particular destinations.

Another problem to be solved by the SNMPv2 effort was related to the management of multiple logical entities via a single agent. In the early days of SNMPv1, many MIB documents were specified which assumed a one-to-one mapping between an agent and an implementation of the MIB. However, technological advances have rendered this assumption invalid. For example, within a modern LAN switching hub, there are often multiple instances of the repeater and bridge MIBs, all accessed via a single agent and thereby requiring a mechanism to de-multiplex the management traffic among the multiple instances of these MIBs. The mechanisms within the SNMPv1 framework for handling this were inadequate.

Finally, many lessons had been learned through implementation and deployment experience, and these lessons led to multiple improvements in the SNMPv2 design. These included introduction of a new operator (`get-bulk`) to allow efficient retrieval of large blocks of data, and larger counters (64-bit) to instrument rapidly increasing variables, such as the octet counters in high speed networks. The SNMPv2 specifications also provide standard transport mappings for use of SNMP over multiple transports, including Novell's IPX, AppleTalk's DDP, and OSI, in addition to the traditional mapping of SNMP over UDP over IP. Finally, the design was updated to reflect many years of experience in the design and specification of MIB documents.

These and other issues identified by the vendors and users of SNMP-based technology were addressed in the initial specifications of the party-based SNMPv2, published in April, 1993, as Proposed Standards [2]. Many expected a wide range of party-based SNMPv2 products to appear shortly thereafter. This did not happen. One key reason is that, though implementable, the party-based security and administrative were often criticized as too difficult for network administrators to deploy, configure, and use. Furthermore, the remote configuration mechanisms resulted in large agent memory requirements which further inhibited acceptance of party-based SNMPv2.

For these reasons, and others, when the SNMPv2 Working Group was reconvened in late 1994 to consider moving the party-based SNMPv2 documents forward in the standards process, many different proposals were put forward for consideration. By the Fall of 1995, the group had decided to explore alternatives to the cumbersome party-based administrative framework, and discussion focused on two simpler user-based approaches, SNMPv2u [5] and SNMPv2*.

Unfortunately, when the allotted time expired, the Working Group had not reached "rough consensus" on which ideas from the two proposals should be accepted into the Draft Standard version of SNMPv2. Rather, the Working Group recommended:

- Publication of the documents where rough consensus had been reached;
- Specification of an interim administrative model based on SNMPv1;
- Uninterrupted continuation of work to complete specification of the remaining unfinished elements.

The first recommendation led to the publication of the SNMPv2 document set as Draft Standards [3].

The second recommendation resulted in the publication of the *Community-based SNMPv2* (SNMPv2C) [4] in which minor modifications to the SNMPv1 security and administrative model are linked to the portions of SNMPv2 which reached Draft Standard status. These portions include the `get-bulk` operator, 64-bit counters, improved MIB specification language, multiple transport mappings, and other improvements.

The unfinished elements consist of:

- Security, including authentication and privacy;
- An administrative framework including contexts, authorization, and access control;
- A remote configuration MIB; and
- Manager-to-manager communications.

As a result, SNMPv2C does not provide the security, administrative framework, remote configuration, and support for multiple logical entities, all of which were key requirements which drove the original development of SNMPv2 in the early 1990s. The need for these features is even greater today.

Although the Working Group was not granted its request that the work continue uninterrupted, both the SNMPv2u and SNMPv2* teams have continued their work independently, each building upon the SNMPv2 documents which are now Draft Standards.

continued on next page

Introduction to SNMPv2* *(continued)*

The following sections provide a technical overview of the core aspects of the SNMPv2* design.

Technical Overview

The SNMPv2* design defines the overall structure of the administrative framework as multiple layers, consistent with good engineering principles. This design follows the basic architecture of a separate authentication service outlined by the SNMPv1 specification [1].

The administrative model defines the aspects which are independent of any particular authentication and privacy service. These include context selection and mechanisms for authorization and access control.

Definitions of services which provide authentication and privacy mechanisms complement the administrative model and constitute another layer. SNMPv2* defines several such services, one providing neither authentication nor privacy, another providing authentication but no privacy, and yet another providing both authentication and privacy.

Appropriate MIB modules provide suitable instrumentation for the administrative model and the authentication and privacy services.

Another document focuses on seamless multilingual support including the integration of SNMPv1 and SNMPv2C into the SNMPv2* administrative framework. This seamlessness is essential for providing good coexistence and ease of transition, both of which are important factors when defining a new version of any infrastructure component which is as pervasive as is SNMP-based management.

SNMPv2* uses the layers specified in existing SNMPv2 specifications whenever possible. At the outermost layer, SNMPv2* uses the transport layering specified in the SNMPv2 document set, including standard mappings for UDP/IP, IPX, DDP, etc. The innermost layer consists of the protocol operations (`get`, `get-next`, etc.) specified in the SNMPv2 document set.

Design Layering

The SNMPv2* administrative model defines mechanisms for authorization, access control, context selection, and other parts of the administrative framework, all of which is common to all authentication and privacy service specifications, which are defined independently. The administrative model also specifies the basic functions all authentication and privacy services must provide (while remaining silent about the particular mechanisms they should use) and specifies the interfaces an authentication and privacy service must support.

The partitioning in the SNMPv2* design reflects the expectation that new authentication and privacy services will be introduced over time, e.g., to accommodate public key technologies or to accommodate advances in cryptography. However, SNMPv2* layering limits the impact of change, because it is architected to accommodate these changes without requiring expensive changes to the stable core of the administrative model. For example, the manner in which logical contexts are named is often used within so-called agent method routines and management station application programming interfaces. Therefore, any change to context naming is destabilizing in that it entails enormous retooling of existing systems. The ability of the stable administrative model to provide extensibility within appropriate boundaries is one of the strengths of the SNMPv2* design.

Administrative Model

In order to provide authorization and access control services, the SNMPv2* administrative model specifies three types of naming:

- Naming of SNMPv2* entities (such as managers and agents).
- Naming of identities (such as users or applications) on whose behalf SNMPv2* messages are sent.
- Naming of the contexts in which management information exists (such as `repeater1` or `bridge3`).

Naming of SNMPv2* entities

Unlike SNMPv1, SNMPv2* requires identification of the entities which exchange SNMPv2* messages. In SNMPv1, naming was typically done with an IP address, such as the IP address found in the SNMPv1 `trap` header. However, this approach was unnecessarily IP-centric and was problematic when used with multi-homed hosts or hosts having dynamic address assignments.

SNMPv2* instead uses an *snmpID*, which is a unique identifier for an entity within a management domain (typically globally as well). In general, this can be thought of as a protocol engine identifier, which uniquely names a manager or agent.

The *snmpID* assists both in the naming of identities and in the naming of management information.

Naming identities

The model specifies that SNMPv2* messages are sent on behalf of an identity, which can be thought of as loosely corresponding to a person or an application. The exact nature of the mapping is dependent upon the authentication and privacy service. For example, SNMPv2* specifies mappings between identities and human-friendly names like "joe" or "openview" as well as mappings between identities and community strings used in SNMPv1 and SNMPv2C.

However, usernames like "joe" and community strings like "public" are not normally unique within an administrative domain, so the identity is simply paired with an *authSnmpID*—an *snmpID* used for authentication and authorization. This pairing of the identity with a node identifier provides the required uniqueness. The authentication and privacy services supply this pair of values to the authorization and access control services when a packet is received. Similarly, the authorization and access control services provide the pair of values to the authentication and privacy services when a packet is to be transmitted.

Naming management information

The third type of naming is that of management information. Naturally, many different instances of the same piece of management information can exist within a given management domain. However, communication between the management system and managed system requires an unambiguous method for naming a single instance of that management information. The context information serves this purpose. In order to uniquely identify management information, four things must be known:

- The identification of the system where the information "lives." The system is named using a *contextSnmpID*—an *snmpID* used for context identification. Note that the system is often a physical device but it may be a logical device, a subset of a physical device, and so on.
- The identification of the logical unit within the system, of which there may be one or more. This is named using a textual string called *contextName,* e.g., "`repeater1.`"
- The name of the object, e.g., `ifSpeed.`
- The object instance, e.g., 1.

continued on next page

Introduction to SNMPv2* *(continued)*

These four pieces of information—*contextSnmpID, contextName,* object name, and object instance—collectively unambiguously name an instance of a piece of management information.

Note further that a context can have a "time value" associated with it. That is, the context can specify the value of a piece of management information in the current running system ("now") or after the next reboot. This is useful when a manager wishes to set or inspect the value that a variable should assume after reboot. This idea is not new to SNMPv2*, but is very much simplified over that which is used by either party-based SNMPv2 or SNMPv2u.

Authorization and Access Control

The SNMPv2* administrative model defines the mechanisms for authorization and access control. Each properly authenticated identity (such as user "joe") is a part of a "group," which is named by a user-friendly string. Group names are site-dependent but could be such things as "guest," "helpdesk," "operator," "shiftsupervisor," and "root." Each of these groups (which may correspond to zero, one, or many human users) is authorized for certain types of operations (such as reading, writing, or none). Similarly, access control is provided according to group membership. For example, the "dba" group might have write access to the variables which allow configuration of the database subsystem whereas other groups might have write access to other variables, but not the database.

When an agent receives a request, it parses the message. Each message header contains a field called *securityFlags,* which is made up of an *sPI* (*security protocol identifier*) and a *reportableFlag* (discussed below). The sPI is used to select which authentication and privacy service to use to authenticate (or authenticate and decrypt) the incoming message.

The designated authentication and privacy service processes the message, authenticating the identity and decrypting the message as appropriate. If the authentication and privacy service is unable to complete its task due to an error condition, suitable error handling is initiated. Otherwise, the authentication and privacy service provides an authentication *snmpID,* identity name, and group name to the authorization and access control service, along with the protocol data unit (PDU), i.e., the body of the request message.

The authorization and access control service of the agent consults the local configuration datastore to determine if the members of the group are authorized to perform the operation (read or write) on the context. If the operation is authorized, it is carried out with respect to the relevant MIB view within that context, during which time the access control policies are enforced and a response is generated.

Readers who are familiar with party-based SNMPv2 will recognize the similarities to "MIB views" which are a part of that design. The primary difference is that SNMPv2* views are simpler and are specified per group rather than per party. The *groupName* provides an easy way of pooling multiple users who can be authenticated independently into a unit with the same access rights. This yields dramatic savings in the amount of configuration required of the administrator and corresponding savings in the amount of non-volatile memory required to store the configuration. It also has the advantage of separating "people" from "policies" thereby making it possible to allow a clerk to add and delete users within a group without giving the clerk access to the parameters that set the access control policies for the various groups.

Multiple sPIs Per Identity

The identity conveyed by a particular message is derived from the combination of the *sPI,* authentication *snmpID,* and an *identityName.* The inclusion of the authentication *snmpID* allows the same "user" to have different passwords on various nodes, such as during a password update. The inclusion of the sPI allows a human manager to configure a single "user" on many different systems but with different authentication and privacy services. This means, for example, that a user can communicate among domestic sites using a strong but non-exportable privacy service while that same user can communicate with foreign sites using a weaker, but exportable, privacy service. This is not a feature, but a requirement, in today's international marketplace.

Reports

There were multiple valuable lessons learned from the otherwise unsuccessful experimentation with the party-based SNMPv2. One of those lessons was that errors often occur when processing messages and that additional feedback to management stations is helpful in diagnosing these conditions. Consequently, a *ReportPDU* was added in the latter stages of the design of the party-based SNMPv2, and SNMPv2* incorporates these same principles.

There are many occasions during processing of SNMPv2* messages when an error may be encountered and processing of the message is halted. In many cases, a suitable error counter (e.g., "unknown sPI" or "unavailable context") is incremented. The SNMPv2* entity may then be required to generate a `report` message to be sent to a logically remote management station which includes a single *VarBind* containing the counter value which was incremented.

Given that it is considered bad design to send error messages in response to error messages, the *reportableFlag* (previously mentioned as being part of every SNMPv2* message header) identifies messages for which a `report` can be generated.

Proxy forwarding operations

An important strength of SNMPv2*'s administrative model is the thorough thought given to proxy forwarding operations. These occur when a system is used to forward SNMP traffic on behalf of other nodes. Such operations often involve "protocol conversion," such as the mapping of one version of SNMP into another, and "transport service mapping" operations, such as the conversion of an SNMP transaction over UDP/IP into an SNMP transaction over IPX. As such, proxy forwarding operations form an important component in the transition and coexistence strategy.

It is true that very many agent systems will only perform local agent operations. For this reason, not only are the administrative MIB objects dealing with proxy agents optional, but the text dealing with proxy agents is highly modularized. Nonetheless, it is clear that there *is* a critical need for proxy agents, particularly when considering the transition period during which SNMPv1, SNMPv2C, and SNMPv2 agents will all participate in the management scheme.

MIB Support

Each SNMPv2* entity maintains a *Local Configuration Datastore* (LCD) which has information about contexts, identities, access rights and so on. Some of this information is remotely accessible as MIB objects defined in the Administrative MIB discussed below. Additional MIB objects are sometimes required by the authentication and privacy services.

continued on next page

Introduction to SNMPv2* *(continued)*

The SNMPv2* designers see an inevitable linkage between the administrative model and the MIB. For example, it is difficult, if not impossible, to write an unambiguous set of procedures referencing elements in the LCD unless the LCD is fully described. The designers model the LCD as a MIB module, using the powerful data description language of the MIB grammar. Therefore, the SNMPv2* designers designed the MIB at the same time the protocols and administrative model were defined. The designers perceive the tight coupling between the design of the model and the MIB to be a strength.

Administrative MIB

The SNMPv2* Administrative MIB provides management objects necessary for remote management of an SNMPv2* entity. The highly modular MIB consists of three required tables and four tables which are optional or conditionally optional, plus some scalars. These tables allow remote access to information about contexts, access control and authorization, MIB views, remote configuration of `inform` and `trap` destinations, as well as proxy and inform parameters.

However, many of the tables are conditionally optional. For example, if a system does not perform proxy forwarding operations or send Inform messages, it obviously is not required to implement that portion of the MIB. The agent implementor has clear and modular choices for which portions of the MIB to implement.

The MIB is designed to be sensitive to those systems with limited non-volatile storage. Only three of the tables are mandatory for all systems, and those three tables may be implemented read-only. User-friendly strings are used for naming entries in the MIB to avoid some of the configuration problems of party-based SNMPv2. In memory-constrained systems, the length of these textual strings may be limited to a single octet. Party-based SNMPv2 had a high requirement for non-volatile storage. SNMPv2* does not repeat that mistake.

User-Based Symmetric Security

The *User-Based Symmetric Security Protocols* specify an initial set of security protocols to be used by SNMPv2* entities.

Threats

As in all previous efforts to define "industrial strength" security for SNMP, the SNMPv2* user-based symmetric security protocols are designed to counter four often-described threats and fulfill four requirements. These are:

- Modification of information (data integrity)
- Masquerade (data origin authentication)
- Message replay (message timeliness)
- Protection from disclosure (privacy)

Heritage

The SNMPv2* user-based symmetric security protocols are nearly identical to those used by SNMPv2u. However, recent updates to SNMPv2u—keyed MD5 and localized keys—have not as yet been incorporated but are under consideration.

Administration of Security

As the name implies, the user-based security protocols center on the concept of a user, each of which is associated with a set of attributes:

- *userName:* the name of the user
- *groupName:* the group to which this user belongs
- *authPrivateKey:* the 16-octet key used by the *usecAuth(5)* service
- *privPrivateKey:* the 16-octet key used by the *usecPriv(6)* service

Additionally, in order to provide for protection against message replay, each SNMPv2* entity is required to maintain two values:

- *snmpBoots:* number of times the SNMPv2* entity has rebooted since its *snmpID* was last reconfigured. This value must be maintained in non-volatile storage.
- *snmpTime:* number of seconds since the SNMPv2* entity last incremented *snmpBoots.*

SNMPv2* AuthInfo

The *authInfo* structure carries the parameters used for authenticating messages. This structure is a part of the header of SNMPv2* messages, which look like the following when user-based symmetric security is being used.

```
SnmpV2Message ::= SEQUENCE {
  version
    INTEGER { version-2(2) },
  mms -- sender's maximum message size
    INTEGER (484..2147483647),
  securityFlags -- includes both the sPI and reportableFlag
    INTEGER (1..2147483647),
  SnmpV2AuthMessage :: [9] SEQUENCE {
    AuthInfo ::= [10] IMPLICIT SEQUENCE { -- if user
                                          -- based security
      authSnmpID
        OCTET STRING (SIZE(12)),
      userName
        OCTET STRING (SIZE(1..32)),
      authSnmpBoots
        Integer32 (0..2147483647),
      authSnmpTime
        Integer32 (0..2147483647),
      authDigest
        OCTET STRING (SIZE(0|16))
      }
      contextSnmpID -- globally unique context id
        OCTET STRING (SIZE(12))
      contextName -- local context information
        OCTET STRING
      pdu
        PlainOrEncryptedPDU
    }
}
```

The Digest Authentication Protocol

User-based symmetric authentication is provided through the use of a digest-based authentication protocol with private keys and a monotonically increasing pair of time indicators.

The first two threats are protected against using the *Digest Authentication Protocol,* which makes use of a message digest algorithm, MD5, documented in RFC 1321. Whenever a message is to be authenticated according to the *sPI* value u*secAuth(5)* or *usecPriv(6),* the user's 16-octet secret key known to both the sender and receiver of the message (*authPrivateKey*) is inserted into the *authDigest* field. The *SnmpV2-AuthMessage* portion of the *SnmpV2Message* is serialized and run through the MD5 algorithm, producing a "fingerprint" or digest of the message. This "fingerprint," also 16 octets, is then inserted into the message "on top" of the *authDigest.*

Upon receipt of the message, the recipient saves the *authDigest* field, inserts the same private authentication key into the message, performs the same MD5 computation, and verifies that the computed digest is the same as that received in *authDigest.*

continued on next page

Introduction to SNMPv2* *(continued)*

If the digests are the same, one can safely conclude that the message was not altered in transit. Furthermore, it authenticates the identity of the user on whose behalf the message was sent. Note further that this authenticated message cannot be replayed to a different SNMPv2* entity since the *authSnmpID* field uniquely identifies that particular SNMPv2* entity. Of course, this mechanism in no way can protect against those circumstances under which the user's password has been compromised.

As an additional test, the SNMPv2* entity supporting *usecAuth(5)* or *usecPriv(6)* can be configured to verify the source transport address from which the message was sent.

Defense against the third threat (message replay) is provided by determining the timeliness of received messages through the use of a pair of time indicators. For example, an agent receiving an SNMPv2* message authenticated with *usecAuth(5)* or *usecPriv(6)* compares its local values for *snmpBoots* and *snmpTime* with the received values of *authSnmpBoots* and *authSnmpTime.*

Continuing this example, if the received value of *authSnmpBoots* is less than the local value of *snmpBoots,* the message is deemed "too old." Similarly, if the values of *authSnmpBoots* and *snmpBoots* are the same, but the value of *authSnmpTime* is more than 150 seconds less than *snmpTime,* the message is deemed "too old." This provides a 150 second window in which messages must arrive.

Of course, if two communicating entities should get out of sync with one another, there are automatic mechanisms in place for resynchronization. Given that the messages exchanged based upon these authentication and privacy services require the existence of shared knowledge, one "end" of the communication must be considered authoritative with regard to that shared knowledge. The non-authoritative pair of time indicators is updated from the authoritative values should the two views become inconsistent.

Symmetric Encryption Protocol

When the *sPI* specifies that privacy is required, i.e., *usecPriv(6),* the serialized PDU is encrypted by the sender and decrypted by the recipient using the *Data Encryption Standard* (DES) in cipher block chaining mode. A shared 16-octet key is used. [11, 12]

User-based Security MIB

The user-based security MIB complements and extends the SNMPv2* administrative MIB by providing instrumentation which is unique to the user-based symmetric authentication and privacy service. It allows managers to add and delete users remotely with great ease, and to update their secrets. It also provides diagnostic counters for various error conditions.

Implementation Status

As of early April, 1996, there are at least three independent and interoperable implementations of SNMPv2*. One of these was developed by Steve Waldbusser based on the CMU implementation and is expected to be made freely available. Commercially supported manager, agent, and mid-level manager implementations are also available.

The SNMPv2* document set is being updated to include improvements based upon recent implementation experiences, interoperability testing, advancements in SNMPv2u, and suggestions from participants in the open SNMPv2* process.

Interoperability Demonstration

A multivendor "SNMPv2* Technology Demonstration" team was formed in mid-March with two principal purposes. First, the team produced a demonstration of SNMPv2* technology at NetWorld+ Interop the first week of April in Las Vegas, Nevada, USA, which showed some of the positive aspects of the SNMPv2* design while serving as a catalyst for interoperability testing of early implementations. Second, the team jointly authored a formal request that the Internet Engineering Task Force (IETF) act now to start the fair and open process leading to a standard for the next generation of SNMP, dubbed *SNMPng*.

The participants in the NetWorld+Interop demonstration included staff members from Bay Networks, BGS Systems, Cisco Systems, Hewlett Packard, International Network Services, and SNMP Research.

These two complementary strategic efforts represent a turning point for secure SNMP-based management, because they both lead toward the goal of having a single acceptable specification addressing customer needs become both an IETF standard and gain widespread market acceptance resulting in a *de facto* industry standard. Strong multivendor support of these two strategic efforts is key because it accelerates progress toward the shared goal of a single standard.

The theme of the demonstration was "Good Ideas Coming Together." This is appropriate for two reasons. First, as the name implies, SNMPv2* (as in DIR *.txt) is a synthesis of the good ideas found in multiple proposals. Second, the request to reconvene the IETF standardization process recommends the development of the next generation of SNMP (SNMPng) based on the best ideas from both SNMPv2* and its leading competitor, SNMPv2u.

Demonstration Details

The SNMPv2* technology demonstration showed agents residing in workstations, switches, routers, hubs, and mid-level managers communicating securely with management stations over InteropNet. [6] All of the implementations showed trilingual behavior, i.e., supporting SNMPv1, SNMPv2C, and SNMPv2*.

The agent implementations included a Bay Networks 100BaseT hub, Cisco C4000 and C2503 routers, HP/UX and Solaris workstations with the HP EMANATE extensible agent system, an Ethernet repeater with RMON, SNMP Research's LATIN LT301 interfaced to RS-232 based legacy devices, and an uninterruptible power supply.

Management station applications included HP OpenView Network Node Manager Release 4.1 (Tornado Release 2), BEST/1 Performance Assurance for Networked Systems and Applications from BGS Systems, and several *Tcl/Tk*-based management applications from SNMP Research, including a remote configuration tool and Mrs. Wright.

Secure manager-to-manager communications were demonstrated by Hewlett Packard and SNMP Research. Hewlett Packard showed their Distributed Internet Technology Remote Data Collectors including secure communications between multiple copies of Network Node Manager. SNMP Research showed secure configuration of a mid-level manager (MLM) using their MLM Configuration Tool.

The demonstration also highlighted remote configuration using SNMPv2* technology with an emphasis on ease-of-use. The demonstration showed use of a GUI-based remote configuration tool for security administration, including the secure configuration of agents and managers.

continued on next page

Introduction to SNMPv2* *(continued)*

This tool makes frequent and routine operations such as adding and disabling users, installing new agents, installing new managers, and related maintenance chores quick and easy to perform. Proving ease-of-use is key because of the past failures of the party-based SNMPv2 in this important area.

It is worthwhile to note that the demonstration showed SNMPv2* implemented in low cost resource-limited embedded systems, including an 80C188-based UPS and an Ethernet repeater, which implemented trilingual SNMPv1, SNMPv2C, and SNMPv2* with IP, UDP, TCP, telnet, MIB2, the Ethernet repeater MIB, and RMON [8] in less than 256 Kbytes of ROM and 512 bytes of non-volatile storage. This shows that SNMPv2* was carefully designed to allow implementation in systems with extremely limited memory assets. Again, this is key because of the past failures of the party-based SNMPv2 in this important area.

The level of vendor involvement in the demonstration, which was put together in only three weeks, approaches if not exceeds the amount of vendor involvement in implementations of the historic party-based SNMPv2 during the two years it was a Proposed Standard. This is important because strong multivendor support accelerates progress toward the shared goal of establishing a single specification which meets customer needs and is both a market standard and an open IETF standard. This is a third key area where SNMPv2* avoids the past failures of the party-based SNMPv2.

The Open Process is Best

Some have called for the next generation of SNMP standards to be set through competitive market forces. While strong multivendor support such as that shown by the SNMPv2* technology demonstration can easily lead to a *de facto* standard, it would not necessarily lead the goal of the establishment of an open IETF standard.

When the market approach is used exclusively:

- There is no guarantee that a single *de facto* standard will ever emerge.
- Competition and market forces often result in a sub-optimally engineered standard.
- It tends to take a long time.
- Until there is convergence on a single set of specifications, either through natural forces and survival of the fittest, or via synthesis, it is unnecessarily and prohibitively expensive for users and vendors to support multiple incompatible technologies.

For these reasons, it is timely and appropriate to restart the IETF SNMP standardization process which is fair and open to all. Consequently, the design team has requested that the IETF act immediately to resume its role. The request recommended formation of an *SNMPng Directorate* to oversee the process and restart the work which has been stalled since last summer.

Conclusion

The timely inclusion of strong security mechanisms and remote configuration into the next generation of SNMP-based management is key to its success. In particular, the requirements resulting from use of SNMP-based management in areas such as systems and application management require the prompt resolution of the issues left unresolved by RFCs 1902–1908.

Both SNMPv2u and SNMPv2* provide a good foundation upon which to complete this work. It is the desire of many of those involved with the SNMPv2* effort that the open process be restarted which would lead to IETF standardization of the missing aspects of the SNMP management framework, including security, an administrative framework, and a suitable remote configuration MIB.

For Further Information

The Internet Drafts specifying SNMPv2* can be retrieved from the usual repositories or can be found via the SNMPv2* Web site located at:

`http://www.snmp.com/v2star.html.`

To subscribe to the open SNMPv2* mailing list send a subscription request to: `snmpv2star-request@snmp.com.`

References

[1] J. Case, J. Davin, M. Fedor, M. Schoffstall, "A Simple Network Management Protocol (SNMP)," RFC 1157, May 1990.

[2] J. Case, K. McCloghrie, M. Rose, S. Waldbusser, Proposed Standard RFCs SNMPv2, RFCs 1441–1452, 1993.

[3] J. Case, K. McCloghrie, M. Rose, S. Waldbusser, Draft Standard RFCs for SNMPv2, RFCs 1902–1908, January 1996.

[4] J. Case, K. McCloghrie, M. Rose, S. Waldbusser, "Introduction to Community-based SNMPv2," RFC 1901, January 1996.

[5] K. McCloghrie, "An Administrative Infrastructure for SNMPv2," RFC 1909, February 1996; G. Waters, "User-based Security Model for SNMPv2," RFC 1910, February, 1996.

[6] B. Krupczak, S. Hultquist, "Managing the InteropNet™," *ConneXions,* Volume 10, No. 5, May 1996.

[7] *ConneXions,* Two *Special Issues* on Network Management and Network Security, Vol. 3, No. 3, March 1989 and Vol. 4, No. 8, August 1990.

[8] Hughes, J., "The RMON Standards for Network Monitoring," *ConneXions,* Volume 8, No. 1, January 1994.

[9] Stallings, W., "RMON2: The Next Generation of Remote Network Monitoring, *ConneXions,* Volume 10, No. 5, May 1996.

[10] Rose, M. T., *The Simple Book: An Introduction to Networking Management,* Revised Second Edition, Prentice-Hall, ISBN 0-13-451659-1, 1996.

[11] Stallings, W., "Cryptographic Algorithms, Part I: Conventional Cryptography," *ConneXions,* Volume 8, No. 9, September 1994.

[12] Stallings, W., "Cryptographic Algorithms, Part II: Public-Key Encryption and Secure Hash Functions." *ConneXions,* Volume 8, No. 10, October 1994

[13] Kaliski, B., "An Overview of Public-Key Cryptography Standards," *ConneXions,* Volume 6, No. 5, May 1992.

DAVID PARTAIN, who works in Linköping, Sweden, is the Managing Director of the European office of SNMP Research International, Inc., a major supplier of SNMP-based products. In addition to software development, Mr. Partain's office is responsible for pre- and post-sales technical support and training for European customers. He can be reached at `partain@europe.snmp.com.`

RMON2

The Next Generation of Remote Network Monitoring

by William Stallings

Introduction

The most important addition to the basic set of SNMP standards (SMI, MIB, SNMP) is the *Remote Network Monitoring* (RMON) specification. RMON is in fact the definition of a management information base (MIB) of variables, or objects, that can be accessed and/or updated via SNMP. The remarkable feature of RMON is that, while it is simply a MIB specification, with no changes in the underlying protocol, it provides for a significant expansion in SNMP functionality.

With most of the MIBs defined for SNMP, the network manager can obtain information that is purely local to individual devices. Consider a LAN with a number of devices on it, each with an SNMP agent. An SNMP manager can learn of the amount of traffic into and out of each device but, without RMON, cannot easily learn about the traffic on the LAN as a whole. Devices that traditionally have been employed to study the traffic on a network as a whole are called *network monitors;* they are also referred to as *network analyzers,* or *probes.* Typically, a monitor operates on a LAN in "promiscuous" mode, viewing every packet on the LAN. The monitor can produce summary information, including error statistics, such as a count of undersized packets and the number of collisions; and performance statistics, such as the number of packets delivered per second and the packet size distribution. The monitor may also store packets or partial packets for later analysis. Filters can be used to limit the number of packets counted or captured, based on packet type or other packet characteristics.

For the purposes of network management in an internetworked environment, there would typically need to be one monitor per subnetwork. The monitor may be a stand-alone device whose sole purpose is to capture and analyze traffic. In other cases, the monitoring function is performed by a device with other duties, such as a workstation, a server, or a router. For effective network management, these monitors need to communicate with a central network management station. In this latter context, they are referred to as *remote monitors.*

A note on terminology: A system that implements the RMON MIB is referred to as an *RMON probe.* The probe has an agent that is no different from any other SNMP agent. It also has an RMON probe process entity that provides the RMON-related functionality. The probe entity is capable of reading/writing the local RMON MIB in response to management action and in performing the various RMON-related functions described in this article. In the literature, you will sometimes see the term *RMON agent.* The term *RMON probe* is preferred.

The evolution of RMON

The RMON MIB specification was first published in 1991 as RFC 1271 [1]. This version of RMON focused on providing statistics and monitoring traffic on MAC-level flows.

The initial RMON MIB is divided into 10 groups of objects, or variables:

- *statistics:* maintains low-level utilization and error statistics for each subnetwork monitored by the agent; limited to Ethernet LANs.
- *history:* records periodic statistical samples from information available in the statistics group.

- *alarm:* allows the person at the management console to set a sampling interval and alarm threshold for any counter or integer recorded by the RMON probe.
- *host:* contains counters for various types of traffic to and from hosts attached to the subnetwork.
- *hostTopN:* contains sorted host statistics that report on the hosts that top a list based on some parameter in the host table.
- *matrix:* shows error and utilization information in matrix form, so the operator can retrieve information for any pair of network addresses.
- *filter:* allows the monitor to observe packets that match a filter. The monitor may capture all packets that pass the filter or simply record statistics based on such packets.
- *packet capture:* governs how data is sent to a management console.
- *event:* a table of all events generated by the RMON probe.
- *tokenRing:* maintains statistics and configuration information for token ring subnetworks.

Each group is used to store data and statistics derived from data collected by the monitor. A monitor may have more than one physical interface and hence may be connected to more than one subnetwork. The data stored in each group represents data gathered from one or more of attached subnetworks, depending on how the monitor is configured for that particular group.

In 1993, RFC 1513 was published, which defines extensions to the RMON MIB for managing IEEE 802.5 token ring networks [2, 3]. Then, the original RMON specification was revised and issued as RFC 1757 in 1995 [4]; the revision makes use of some of the MIB conventions for SNMPv2, but remains compatible with SNMPv1.

Finally, in 1994, work began on an extension to the RMON MIB to include monitoring of protocol traffic above the MAC level. This work, which is referred to as RMON2, has resulted in two Internet Drafts that will soon be published as RFCs [5, 6].

RMON2

Using the original RMON, now referred to as RMON1, an RMON probe can monitor all of the traffic on the LANs to which it is attached. It can capture all of the MAC-level frames and read the MAC-level source and destination addresses in those frames. The probe can provide detailed information about the MAC-level traffic to and from each host on each attached LAN. However, if a router is attached to one of these LANs, the RMON1 probe can only monitor the total traffic into and out of that router; it has no way of determining the ultimate source of incoming traffic arriving via the router or the ultimate destination of outgoing traffic leaving via the router.

With RMON2, the RMON probe now has the capability of seeing above the MAC layer by reading the header of the enclosed network-layer protocol, which is typically IP. This enables the probe to analyze traffic passing through the router to determine the ultimate source and destination.

continued on next page

RMON2 *(continued)*

With this capability, the network manager can answer a number of new questions, such as:

- If there is excessive load on the LAN due to incoming router traffic, what networks or hosts account for the bulk of that incoming traffic?
- If a router is overloaded because of high amounts of outgoing traffic, what local hosts account for the bulk of that outgoing traffic, and to what destination networks or hosts is that traffic directed?
- If there is a high load of pass-through traffic (arriving via one router and departing via another router), what networks or hosts are responsible for the bulk of this traffic?

With answers to questions such as these, the network manager may be able to take steps to contain traffic loads and improve performance. For example, the network manager can see what clients are talking to what servers and place systems on the appropriate network segments to optimize traffic flow.

An RMON2 probe is not limited to monitoring and decoding network-layer traffic. It can also view higher-layer protocols running on top of the network-layer protocol. In particular, an RMON2 probe is capable of seeing above the IP layer by reading the enclosed higher-level headers such as TCP, and viewing the headers at the application protocol level. This allows the network manager to monitor traffic in great detail.

With RMON2, a network management application can be implemented that will generate charts and graphs depicting traffic percentage by protocols or by applications. Again, such a level of detail is useful in containing load and maintaining performance.

It is important to note that in RMON2 terms, any protocol above the network layer is considered "application level."

RMON2 Groups

Figure 1 shows the overall structure of the RMON MIB. All of the RMON1 and RMON2 groups of objects are directly subordinate to an RMON entry. Figure 2 highlights some of the key objects in each of the RMON2 groups.

The *Protocol Directory group* provides a way for an RMON2 manager to learn which protocols a particular RMON2 probe interprets. This information is especially important when the manager and probe are from different vendors. On any particular network, there may be many different protocols running. Some are standardized or at least well-known, while others may be custom protocols developed for a particular application of product. Since the most of the objects in the RMON2 MIB deals with monitoring the activity of these protocols, some common framework is needed to support them all. This is the purpose of the Protocol Directory group, which provides a single central point for storing information about types of protocols. This group consists of a table in which the probe lists one entry for each protocol for which it can decode and count protocol data units (PDUs). The table covers MAC, network, and higher-layer protocols.

The *Protocol Distribution group* summarizes how many octets and packets have been sent by each of the protocols supported. The group includes a table with one row for each protocol defined in the Protocol Directory group. Each row counts the number of packets and the number of octets observed by the probe for a particular protocol.

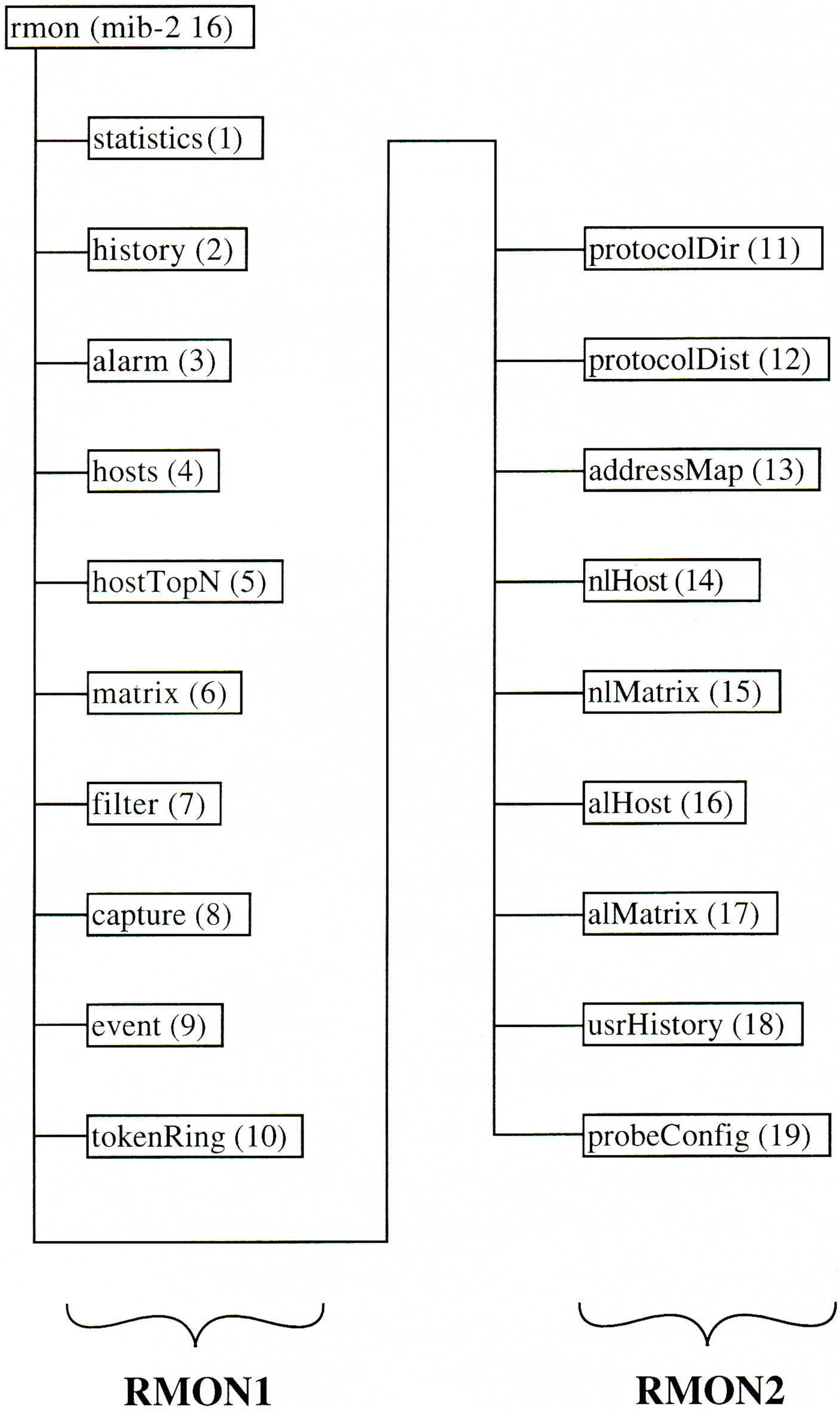

Figure 1: The Remote Network Monitoring (RMON) MIB

The *Address Map group* matches each network address to a specific MAC-level address, and therefore to a specific port on the network device. This is helpful in node discovery and network topology applications for pinpointing the specific paths of network traffic. This group contains a table that is indexed by network (IP) address. Given an IP address, a manager can look up the associated MAC address in the table. The probe builds the table by observing source MAC and network addresses in packets that it sees.

continued on next page

RMON2 *(continued)*

Protocol Directory •Protocol ID •Protocol Parameters	Protocol Distribution •Packets •Octets	Address Map •Network address •Physical address	Network-Layer Host •In packets/host •Out packets/host •In octets/host •Out octets/host	Network-Layer Matrix •Packets/connection •Octets/connection •Top packet rate •Top octet rate

Application-Layer Host •In packets/app •Out packets/app •In octets/app •Out octets/app	Application-Layer Matrix •Packets/connection •Octets/connection •Top packet rate •Top octet rate	User History •Object samples	Probe Configuration •Configuration parameters

Figure 2: RMON2 Groups with Selected Variables

The purpose of the *Network-Layer Host group* is to collect basic statistics on traffic into and out of each discovered host, broken down by network-layer address. This is similar to the RMON1 Host group, which gathers statistics based on MAC address. The Network-Layer Host group lets the network manager look beyond a router to the connected hosts.

This group includes a table with one row for each known network-layer address for each supported network-layer protocol (usually just IP). The probe adds an entry to the table for each new address seen as a source or destination network-layer address in a packet. The table counts the number of packets and the number of octets into and out of each host. Thus, the table gives the manager a breakdown of the load generated on a LAN by source and destination, regardless of whether the source or destination is actually on this LAN.

The *Network Layer Matrix group* provides statistics on the basis of network-layer traffic between pairs of hosts. In a sense, this group collects connection-oriented statistics whereas the Network-Layer Host group focuses on the total traffic into and out of a single host. There are two key sets of data stored in this group. One set is can be viewed as a matrix that stores statistics on traffic between each possible source/destination pair. For each pair, a count is kept of the number of packets and the number of octets transmitted in each direction. The other set of data is referred to as a *TopN* table. The TopN table is a list that ranks pairs of hosts based on either the count of packets or count of octets between the pair in one direction. Thus, the first entry in the table identifies the source/destination pair with the greatest amount of packet or octet traffic observed by the probe.

The *Application-Layer Host group* is similar to the Network-Layer Host group, except that in this case, there is one entry in *alHostTable* for each application-level protocol discovered at each known network-layer address. Keep in mind that, for RMON2, the term "application level" refers to all protocols above the network layer. The Application-Layer Host group includes a table that contains one entry for each application-layer protocol at each network-layer address. Thus, the statistics collected in this group are broken down by application by host. In contrast, the Protocol Distribution group counts the aggregate traffic generated by a protocol for all hosts combined.

The *Application-Layer Matrix group* is similar to the Network-Layer Matrix group, except that again, statistics are collected on the basis of each application-level protocol discovered at each known network-layer address. There is a matrix data set that counts traffic broken down by application-layer protocol between pairs of hosts and a TopN data set that ranks pairs of hosts by application-layer traffic.

The *User History group* is used to define sampling functions for a probe. The probe periodically samples manager-defined variables and logs the data in this group. With this group, the network manager can configure history studies of any counter in the system, such as a specific history on a particular file server or router-to-router connection. The group can be configured to record the value of a particular counter at a given sampling rate. Subject to a maximum table size, the probe keeps all past values that have been sampled. This enables a network management application to analyze trends in a given parameter.

The *Probe Configuration group* is designed to enhance interoperability among RMON probes and managers by defining a standard set of configuration parameters for probes. This makes it easier for one vendor's RMON application to be able to remotely configure another vendor's RMON probe.

Time filtering

An important new capability introduced with RMON2 is *time filtering*. A common function of a network management application is to periodically poll all probes subordinate to it for the values of objects maintained at the probe. For the sake of efficiency, it is desirable to have the probe return values only for those objects whose values have changed since the last poll. There is no direct way in the SNMP or SNMPv2 protocol to achieve this function. However, the RMON2 designers have come up with an innovative means of achieving the same functionality in the MIB definition.

In essence, a table in RMON2 can be equipped with a timestamp object, such that associated with each row in the table is a timestamp that records the last time the row was altered. When an SNMP manager makes a request for rows from this table, it includes in the request a time filter value for each requested row. The probe only returns the value of a particular row if that row has been updated since the time filter value. For example, consider a data table at a probe that currently has two rows:

timeStamp	Protocol Type	Packets
6	1	5
8	2	9

This is a table that counts the total number of packets generated by each known protocol. Currently there are two protocols being monitored. When the probe receives a time-filtered request for a particular row, the prove filters the row as follows:

```
if (timestamp-for-this-row ≥ TimeFilter-value-in-Request)
          /* return this instance in a response PDU */
  else    /* skip this instance */
```

continued on next page

RMON2 *(continued)*

Suppose that a manager had sampled this table at time 7 and received the counter value for both protocol type 1 and type 2. At some later time, the manager sends a request for both rows with a time filter of 8. The probe will only return the data for protocol type 2. As a result, transmission capacity is saved.

Summary

RMON2 extends the capability of the original RMON MIB to include protocols above the MAC level. The inclusion of network-layer protocols, such as IP, enables a probe to monitor traffic through routers attached to the local subnetwork. The probe can therefore monitor the sources of off-network traffic arriving by a router and the destinations of off-network traffic that leaves by a router. The inclusion of higher-layer protocols, such as those at the application level, enables the probe to provide a detailed breakdown of traffic on the basis of application.

References

[1] Waldbusser, S., "Remote Network Monitoring Management Information Base," RFC 1271, November 1991.

[2] Waldbusser, S., "Token Ring Extensions to the Remote Network Monitoring MIB," RFC 1513, September 1993.

[3] Hughes, J., "The RMON Standards for Network Monitoring," *ConneXions,* Volume 8, No. 1, January 1994.

[4] Waldbusser, S., "Remote Network Monitoring Management Information Base," RFC 1757, February 1995.

[5] Waldbusser, S., "Remote Network Monitoring MIB Version 2," Internet Draft (work in progress), January 1996.

[6] Bierman, A, and Iddon, R., "Remote Network Monitoring MIB Protocol Identifiers," Internet Draft (work in progress), January 1996.

[7] M. T. Rose and K. McCloghrie, "Structure and Identification of Management Information for TCP/IP-based internets," RFC 1155, May 1990.

[8] K. McCloghrie, M. Rose, "Management Information Base for Network Management of TCP/IP-based internets: MIB-II," RFC 1213, March 1991.

[9] Marshall T. Rose, "Network management: Status and challenges," *ConneXions,* Volume 7, No. 6, June 1993.

[10] *ConneXions,* Two *Special Issues* on Network Management and Network Security, Vol. 3, No. 3, March 1989 and Vol. 4, No. 8, August 1990.

[11] Case, J. D., McCloghrie, K., Rose, M. T. & Waldbusser, S. L., "Introduction to version 2 of the Internet-standard Network Management Framework," RFC 1441, April 1993.

[This article is based on material in Bill Stallings' *SNMP, SNMPv2, and RMON: Practical Network Management, Second Edition,* ISBN 0-201-63479-1, 1996 by Addison-Wesley. Used with permission. —Ed.]

WILLIAM STALLINGS is an independent consultant whose clients have included major corporations and government agencies in the United States and Europe. He is the author of over a dozen books on data communications and computers, including the forthcoming *Data and Computer Communications, Fifth Edition,* from Prentice-Hall. His e-mail address is: `ws@shore.net`

Call for Participation

The second *IEEE Symposium on Planning and Design of Broadband Networks* will be held at Le Chateau Montebello, Montebello, Quebec, Canada on October 17–20, 1996. The first Symposium, held in October 1994, was an outstanding success with representation from North America, Europe and Asia.

Purpose

The purpose of this symposium is to provide an environment for the discussion and exchange of ideas concerning computer-aided planning and design techniques and tools for broadband networks. The symposium will include both invited and contributed talks, panel discussions, and demonstrations of broadband network planning and design tools. This year, the symposium will also include a strong focus on the application of design tools and contributors are encouraged to display their applications at the exhibition. Abstracts of all presentations will also be distributed at the symposium.

Topics

The symposium will address topics in the following areas:

- Challenges in broadband network planning and design
- Simulation methodologies and tools for planning and design of broadband networks
- Tool applications and deployment case studies

Submissions

Please submit *by June 30, 1996,* 5 copies of the abstract of proposed talk or demos to the Technical Program Chairman:

Professor Hussein Mouftah
Department of Electrical and Computer Engineering
Queen's University
Kingston, Ontario K7L 3N6
CANADA
Telephone: +1 (613) 545-2934,
Fax: +1 (613) 545-6615
E-mail: `mouftah@eleceng.ee.queensu.ca`

More information

For further information please contact:

Ihor Gawdan, Symposium Chair
Nortel Technology
P.O. Box 3511
Station C
Ottawa, Ontario K1Y 4H7
CANADA
Telephone: +1 (613) 763-9926
Fax: +1 (613) 763-2976
E-mail: `igawdan@bnr.ca`

Call for Papers

The *Third International Symposium on Autonomous Decentralized Systems* (ISADS 97) will be held April 9–11, 1997 in Berlin, Germany.

Scope

Driven by the increasing power, intelligence, reliability, and openness of computer, communication and control technologies, a new generation of distributed systems is emerging, that will be able to support distributed business and control applications with extreme efficiency, reliability and security requirements. Such systems are expected to consist of largely autonomous, decentralized and geographically dispersed components interacting via communication networks, and are thus called *Autonomous Decentralized Systems* (ADS). ISADS 97 will primarily focus on advancements and innovations in ADS platforms and applications. Integration of telecommunication and computing aspects into a uniform concept for providing an open distributed processing environment is a key factor.

Topics

ISADS invites papers and panel proposals on the topic of the symposium that will foster interactions among researchers and practitioners in computer, (tele)communication, management, control and other related fields from academia, industry and government. The scope of ADS encompasses but is not limited to:

- Telecommunication information networking architecture
- Distributed system development and maintenance
- Distributed control and its supporting platforms
- Object management architecture/Application frameworks
- Platform and application interoperability
- Reference models for ADS
- Computer-supported cooperative work/Virtual enterprise
- Legacy system integration
- Novel applications of ADS: manufacturing systems, realtime environments, office automation, traffic and transportation control, electronic commerce, etc.

Submissions

Send 6 copies of an original full paper (12 point, double-spaced) with 3,000–6,000 words—in paper form only—to the address given below. Papers should include: title, authors, affiliations, 150-word abstract and list of keywords. Identify the author responsible for correspondence, including the author's name, position, mailing address, telephone and fax numbers, and e-mail address. One of the authors of each accepted paper must present the paper at ISADS 97. Panel proposals should include: title, organizer's affiliations, position, mailing address, telephone and fax numbers, e-mail address, and 150-word scope statement, proposed chair and panelists. The proceedings of the symposium will be published by IEEE Computer Society Press.

Jürgen Nehmer, ISADS 97 Program Chair
Universitaet Kaiserslautern
Fachbereich Informatik
Postfach 3049
D-67653 Kaiserslautern
GERMANY
E-mail: `nehmer@informatik.uni-kl.de`
Phone: +49-631-205-4020 • Fax: +49-631-205-3558.

Important dates

July 15, 1996: All paper and panel proposals due

October 1, 1996: Proposals and authors notified of acceptance

December 2, 1996: Camera-ready copies due

More information

`http://www.fokus.gmd.de/ws/isads97/`

Future NetWorld+Interop Dates and Locations

NetWorld+Interop 96	Frankfurt, Germany	June 10–14, 1996
NetWorld+Interop 96	Tokyo, Japan	July 22–26, 1996
NetWorld+Interop 96	Atlanta, GA	September 16–20, 1996
NetWorld+Interop 96	Paris, France	October 7–11, 1996
NetWorld+Interop 96	London, England	Oct. 28–Nov. 1, 1996
NetWorld+Interop 96	Sydney, Australia	November 25–29, 1996
NetWorld+Interop 97	Singapore	April 7–11, 1997

All dates are subject to change.

More information

Call 1-800-INTEROP or +1-415-578-6900 for more information. Or send e-mail to `info@interop.com` or fax to +1-415-525-0194. For the latest information about NetWorld+Interop including *N+I Online!* as well as other SOFTBANK produced events, check our home page at `http://www.interop.com`

NetWorld+Interop is produced by SOFTBANK Exposition and Conference Company, 303 Vintage Park Drive, Foster City, California 94404–1138, USA.

NETWORLD+INTEROP 96

Write to *ConneXions*!

We'd love to hear your comments, suggestions and questions about anything you read in *ConneXions*. Our editorial address is given below. Use it for letters to the Editor, requests for the index of back issues, questions about particular articles etc.:

ConneXions—The Interoperability Report
303 Vintage Park Drive
Suite 201
Foster City
California 94404–1138
USA
Phone: +1 415-578-6900 or 1-800-INTEROP (Toll-free in the USA)
Fax: +1 415-525-0194
E-mail: `connexions@interop.com`
URL: `http://www.interop.com`

Subscription information

For questions about your subscription please call our customer service hotline: 1-800-575-5717 or +1 610-892-1959 outside the USA. This is the number for our subscription agency, Seybold Publications. Their fax number is +1 610-565-1858. The mailing address for subscription payments is: P.O. Box 976, Media, PA 19063–0976.

CONNEXIONS
303 Vintage Park Drive
Suite 201
Foster City, CA 94404-1138
Phone: 415-578-6900
FAX: 415-525-0194

ADDRESS CORRECTION
REQUESTED

FIRST CLASS MAIL
U.S. POSTAGE
PAID
SAN JOSE, CA
PERMIT NO. 1

CONNEXIONS

EDITOR and PUBLISHER — Ole J. Jacobsen